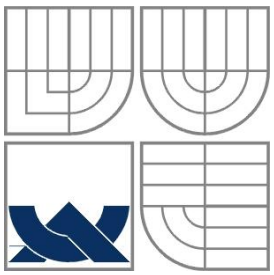
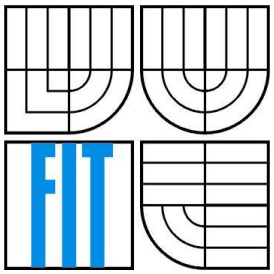


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍ SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

PREDIKCE PROTEINOVÝCH DOMÉN

PROTEIN DOMAINS PREDICTION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MARTIN VALENTA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. IVANA BURGETOVÁ, Ph.D.

BRNO 2013

Abstrakt

Práce představuje oblast proteinů a jejich domén. Stručně popisuje metody získání proteinové struktury na různých úrovních hierarchie. Pozornost je dále věnována existujícím nástrojům pro predikci proteinových domén a databázím sdružujícím informace o doménách. Dále jsou představeni vybraní zástupci metod predikce pracující jak s informacemi o vnitřní struktuře molekuly, tak se sekvencí aminokyselin. V příslušné kapitole je nastíněn realizovaný postup predikce hranic domén z primární struktury proteinu, využívající neuronovou síť. Zbývající část práce se zabývá testováním vytvořené implementace metodou křížové validace a možnostmi jejího dalšího rozvoje.

Abstract

The work is focused on the area of the proteins and their domains. It also briefly describes gathering methods of the protein´s structure at the various levels of the hierarchy. This is followed by examining of existing tools for protein´s domains prediction and databases consisting of domain´s information. In the next part of the work selected representatives of prediction methods are introduced. These methods work with the information about the internal structure of the molecule or the amino acid sequence. The appropriate chapter outlines applied procedure of domains´ boundaries prediction. The prediction is derived from the primary structure of the protein, using a neural network The implemented procedure and its possibility of further development in the related thesis are introduced at the conclusion of this work.

Klíčová slova

Proteiny, struktura proteinů, proteinové domény, klasifikace proteinových domén, databáze proteinových domén, Pfam, SCOP, DOMpred, InterProScan, DoBo, PPRODO, DOMpro, PUU, Domain parser, PSI-BLAST, neuronová síť, Back-Propagation, genetický algoritmus.

Keywords

Proteins, protein structure, protein domain, classification of protein domain. database of protein domains, Pfam, SCOP, DOMpred, InterProScan, PPRODO, DOMpro, PUU, Domain parser, PSI-BLAST, neural network, Back-Propagation, genetic algorithm.

Citace

Valenta Martin: Predikce proteinových domén, diplomová práce, Brno, FIT VUT v Brně, 2013

Predikce proteinových domén

Prohlášení

Prohlašuji, že jsem tento semestrální projekt vypracoval samostatně pod vedením p. Ing. Ivany Burgetové Ph.D.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Martin Valenta

Poděkování

Na tomto místě bych chtěl poděkovat vedoucí práce za její cenné rady, které mi umožnily tuto práci zdárně dokončit.

Obsah

1 Úvod

Predikce proteinových domén spadá do oblasti strukturální bioinformatiky, kde představuje identifikace těchto útvarů jeden z nejdůležitějších kroků při stanovení struktury a s ní spojené funkce proteinové makromolekuly. Domény jsou evolučně silně konzervované útvary s funkcí a strukturou nezávislou na přítomnosti v konkrétním proteinu. Jejich význam by se tedy dal volně přirovnat k jakýmsi genetickým stavebním modulům. Specifické vlastnosti jednotlivých domén byly rovněž využity k zavedení systematické klasifikace struktury proteinů, jejímž významným příkladem je databáze SCOP.

V následující práci se nejprve pokusím vymezit pojem proteinů a jejich domén. Dále velmi stručně představím vybrané laboratorní metody, s jejichž pomocí lze získat informace o struktuře proteinů na různých úrovních hierarchie. Převážnou část textu bude ale věnovat výhradně oblasti predikce proteinový domén.

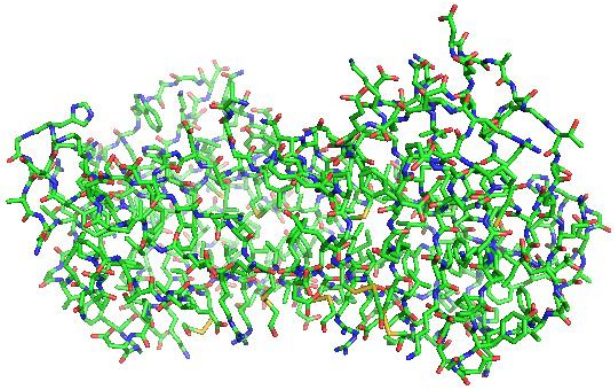
Nejprve se zaměřím na vybrané nástroje volně dostupné ve formě webových serverů a na databáze proteinových domén, které jsou v oblasti predikce často využívány jako zdroje testovacích dat. U jednotlivých zástupců se pokusím shrnout možnosti, které poskytují uživateli, a provedu jednoduchý test jejich funkce.

Na předchozí část pak navážu detailnějším rozborem vybraných postupů predikce, které pracují jak s informací o prostorovém uspořádání molekuly proteinu, tak především s informací o řetězci aminokyselin. Z metod využívajících primární strukturu proteinu se budu věnovat pouze těm zástupcům, kteří využívají postupů strojového učení k vyhledávání specifických oblastí sekvence aminokyselin, nazývaných hranice proteinových domén.

Zbývající část práce věnuji implementovanému postupu predikce proteinových domén, který využívá informace obsažené v profilu multisekvenčního zarovnání metody PSI-BLAST, k detekci výše zmíněných hraničních oblastí pomocí aplikace neuronové sítě. Nejprve popíšu návrh tohoto postupu, který vychází z [18]. Následně představím jednotlivé navrhované součásti a budu komentovat jejich implementovanou podobu. V další části se zaměřím na postup a výsledky testování realizovaného nástroje, které je založeno na metodě křížové validace. V úplném závěru své práce nastíním cesty možného dalšího rozvoje.

2 Obecně o proteinech

Na úvod jsem se rozhodl shrnout některé základní vlastnosti proteinových molekul. Informace k této části jsem čerpal především z učebního materiálu [1].



Obrázek 2.1: Ukázka struktury malého proteinu Gal – 1, vygenerovaná programem PyMOL

Proteiny neboli bílkoviny tvoří většinu suché hmotnosti buňky, kde obstarávají široké spektrum funkcí jako stavební, strukturní, transportní, enzymatickou, regulační a další. Tato rozmanitost pramení z obrovského množství tvarů (konformací), které mohou molekuly zaujímat v prostoru. Základní vlastností proteinové molekuly je schopnost vázat se na určitou jinou konkrétní molekulu – ligand. Tento proces probíhá v jedné z nejdůležitějších částí proteinu – ve vazebném místě. Proteinová molekula se váže pouze na několik málo ligandů. Zmíněná vlastnost je zajištěna tvarem samotného vazebného místa a dala by se přirovnat k zasouvání klíče do zámku. Pouze pokud daný ligand "pasuje" do daného vazebného místa, je vytvořen dostatečný počet slabých vodíkových vazeb nutných k jeho dalšímu udržení.

Z předchozí části jasně vyplývá skutečnost, že dominantní roli pro funkci proteinové molekuly nese její prostorové uspořádání, na které bych se rád nyní zaměřil, protože je přímo spojené s proteinovými doménami. Toto uspořádání je do značné míry určeno vnitřní strukturou proteinu, která je tvořena unikátním (pro konkrétní druh molekuly) řetězcem aminokyselin spojených kovalentní peptidovou vazbou (nazývanou polypeptidová kostra). Ke kostře jsou pak napojeny tzv. postranní řetězce aminokyselin, které daným aminokyselinám propůjčují jejich vlastnosti. Velký vliv na celkovou strukturu má především uspořádaní aminokyselin s polárními a nepolárními postranními řetězci (umožňující ve vodném prostředí shlukovat hydrofobní AK uvnitř kompaktních globulárních útvarů). Celková struktura proteinu je dále dotvářena nekovalentními vazbami mezi různými

kombinacemi atomů kostry a postranních řetězců (např. přítomnost cysteinu umožňuje vznik disulfidických můstků). I přes takto komplikovaný proces utváření finálního prostorového uspořádání molekuly proteinů nabývají jediné stabilní konformace – tj. stavu s nejnižší volnou energií daného právě kombinací aminokyselin v polypeptidovém řetězci [1]. Metoda, která by však umožnila dokonalý přímý překlad sekvence aminokyselin na její přesnou prostorovou strukturu, není v současné době známa.

Pro snazší pochopení výsledné struktury proteinu se zavedlo několik úrovní organizace, pro některé z nich je typické, že se podařilo objevit množství strukturních vzorů, které se hojně opakují v rámci většiny proteinových molekul.

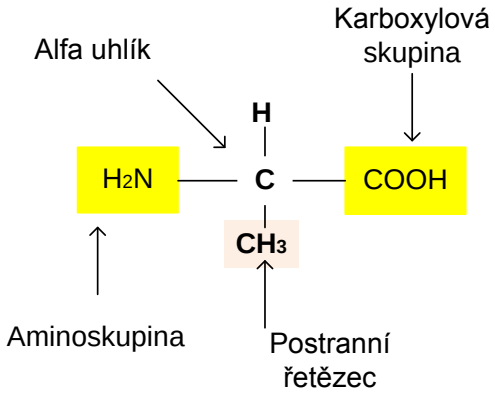
- Primární struktura, která je tvořena přímo sekvencí aminokyselin v rámci proteinové molekuly.
- Sekundární struktura je tvořena dvěma častými strukturními motivy α-šroubovicemi a β-skládanými listy. Tyto motivy vznikají z přítomnosti vodíkových můstků mezi skupinami N-H a C=O v rámci polypeptidové kostry (proto není tedy nutná přítomnost určitého postranního řetězce) a mohou tak být tvořeny různými aminokyselinovými sekvencemi.
- Super-sekundární struktura je tvořena proteinovými doménami představujícími část polypeptidového řetězce, který se dokáže autonomně svinout do kompaktní stálé struktury.
- Terciární struktura popisuje trojrozměrné uspořádání molekuly.
- Kvartérní struktura, je celková struktura proteinů tvořených více než jedním polypetidovým řetězcem.

2.1 Základní představení aminokyselin

Pro účely následujících textu budu uvažovat existenci 20 alfa-L-aminokyselin z celkového počtu 22 (není uvažován selenocystein a pyrolisin). Vybraných 20 kyselin se často označuje jako tzv. proteinogenní AK, zbylé dvě se vyskytují velmi vzácně. Společným znakem všech aminokyselin, který je mimo jiné předurčuje k použití v rámci živých struktur, je jejich rozpustnost v polárních rozpouštědlech (např. vodě). Bližší informace o vlastnostech jednotlivých kyselin a jejich postranních řetězců lze nalézt v příslušné kapitole v [1].

Aminokyselina se skládá z několika základních částí:

- Alfa uhlík, na který jsou napojeny zbylé části a jehož prostorové umístění v rámci molekuly tvoří základní informaci pro činnost některých metod predikce popsaných v dalším textu.
- Postranní řetězec, který určuje chemické vlastnosti celé aminokyseliny a podílí se na finální konformaci proteinu.
- Aminoskupina a karboxylová skupina tvoří tzv. polypeptidovou kostru. Vazba mezi dvojicí aminokyselin pak probíhá napojením aminoskupiny jedné kyseliny na karboxylovou skupinu druhé kyseliny.



Obrázek 1.2: Schematické znázornění aminokyseliny z [1].

2.2 Laboratorní metody umožňující získat informace o vnitřní struktuře proteinů

V následující části textu se pokusím přiblížit několik laboratorních principů umožňujících získat informace o vnitřní struktuře a prostorovém uspořádání proteinových molekul. S ohledem na téma této práce můžu konstatovat, že první metody pro stanovení struktury proteinu (a s ní spojených proteinových domén) využívaly následujících přístupů v kombinaci s anotacemi vloženými odborníkem v oboru strukturní biologie (vybaveného spektrem podpůrných výpočetních nástrojů).

Určování primární struktury – sekvence aminokyselin proteinu

K tomuto účelu se využívají tzv. sekvenační metody. V porovnání s metodami pro určení terciární struktury jsou tyto metody méně náročné – z toho mimo jiné plyne, že větší objem dosud získané informace o proteinech má charakter jejich primární struktury. Metody tohoto typu by se daly dále rozdělit do následujících kategorií.

- Původní metody, které pracovaly přímo s proteinovými molekulami, převzato z [2].
    - *Určování koncové aminokyseliny* - za pomocí reakce dansyl chloridu s N-koncovou aminoskupinou (produkty této reakce slouží jako fluorescenční značka), hydrolízy (vymytí zbytku proteinu) a následné chromatografie.
    - *Edmanova degradace* - využívá procesu postupného odštěpování degradované koncové AK a její následné identifikace. Tento proces je dnes plně automatizován, k činnosti metody je nutné dostatečné množství čistého proteinu. Doba jedné sekvenace se pohybuje kolem 45 minut. Jedná se o alternativu k metodě následující.
    - *Sekvenování* pomocí hmotnostní spektrometrie - využívá metodu tandemové hmotnostní spektrometrie, kdy dochází k dvojici spektrometrických analýz oddělených disociací (rozpadem) vstupních iontů, jejichž výsledkem je hmotnostní spektrum popisující vztah rodičovského iontu a jeho fragmentů (strukturní informace o rodičovském iontu).
- S rozvojem sekvenace DNA se přešlo k tzv. nepřímé sekvenaci – kdy je převedeno pořadí nukleotidů z DNA na sekvenci aminokyselin.

Určování terciární struktury

Metody určující prostorovou strukturu proteinů jsou náročnější než předchozí. K jejich provedení je nutné nejprve připravit krystal proteinu, resp. dostatečný objem vysoce čistého proteinu. Prováděný experiment jako celek poté využívá poměrně sofistikované metody.
Velké množství takto prozkoumaných proteinových struktur lze nalézt v databázi Protein Data Bank. Převzato z [3].

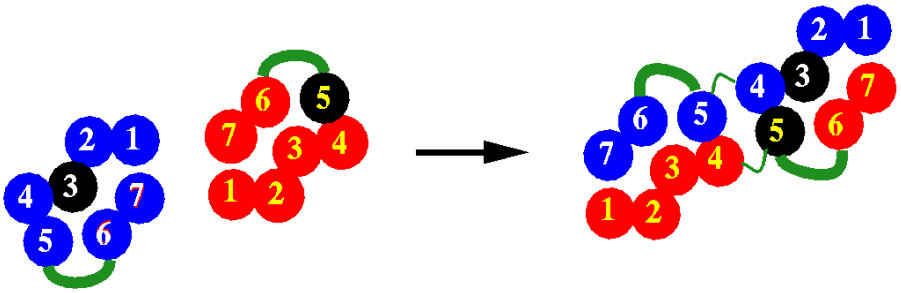
- *Rentgenová krystalografie* – dominantní metoda co se týká objemu prozkoumaných dat (v rámci databáze Protein Databank je tímto způsobem zpracováno přibližně 90% proteinů). Metoda využívá měření intenzity a odchylky rentgenových paprsků po jejich průchodu krystalem proteinu, tento krystal je umístěn v goniometru a postupně rotován. Popsaným způsobem je získána sada dvourozměrných snímků, jejichž konverzí je následně vytvořena trojrozměrná mapa hustoty elektronů v krystalu. Z této mapy lze vyčíst průměrné pozice atomů proteinu a jejich chemické vazby.
- *NMR – nukleární magnetická rezonance bílkovin*. Metoda využívá více rozměrnou magnetickou rezonanci, jejímž principem je měření tzv. jaderného spinu (atomů C, nebo H) resp. jeho deexcitace z excitovaného stavu vyvolaného kombinací velmi silného magnetického pole a radioaktivního záření působícího na zkoumaný substrát. K realizaci těchto experimentů je na rozdíl od předchozího procesu nutné větší množství (300-600 mikrolitrů) koncentrovaného velmi čistého proteinu. Hlavním praktickým omezením této metody je, vyjma její technologické náročnosti, použitelnost pouze na poměrně malé proteinové molekuly.

2.3 Proteinová doména

Proteinové domény jsou částečně nezávislé jednotky struktury proteinu často nesoucí specifickou funkcionalitu. Přesněji řečeno, proteinová doména by se dala charakterizovat jako nezávislá pod-jednotka proteinu s kompaktní strukturou, nesoucí specifickou funkci a procházející nezávisle na zbytku proteinu procesem skládání. Dále je označována jako základní stavební kámen molekulární evoluce. Domény jsou základními elementy terciární struktury proteinu a jako takové se často využívají k porovnávání struktury proteinů.

Domény vznikají kombinacemi prvků sekundární struktury – tzv. motivů. Každá doména obsahuje jádro (relativně pevného skupenství) tvořené hydrofobními prvky sekundární struktury - obklopenými prvky hydrofilními, vytvářejícími kapalný obal. Sekvence kódující jádro domény je často evolučně konzervovaná v rámci proteinových rodin. Velikost domény je limitována v rozmezí 36-692 aminokyselin. Převážná většina domén (cca. 90%) obsahuje kolem 200 residuí. Domény s velikostí větší než 400 residuí obsahují často více hydrofobních jader, což je obvykle dáno procesem jejich vzniku (sloučením několika jednodušších domén, nebo motivů sekundární struktury v rámci evoluce proteinů). Zajímavou vlastností charakterizující doménu jako celek je, že interakce (vazebné) probíhají s vyšší intenzitou mezi částmi v rámci domény než mezi doménou a zbytkem proteinu.

Během procesu dotváření výsledné trojrozměrné struktury proteinu – tzv. skládání mají domény výrazný vliv na jeho průběh. Jejich přítomnost zvyšuje významně rychlost tohoto děje tím, že redukují počet možných interakcí mezi aminokyselinovými zbytky (omezují tím objemný prostor všech možných konformací, kterých může molekula nabývat). Celková struktura a s ní spojené chemické vlastnosti proteinu jsou ovlivněny jak vazebnými interakcemi v rámci domén, tak těmi, ke kterým dochází mezi doménami. Kromě vlivu na skládání proteinu má přítomnost domén dopad na utváření ještě komplexnějších útvarů v rámci kvartérní struktury, která vzniká propojením jednotlivých polypeptidových řetězců do jediné oligomerní molekuly. Mechanismus, který umožňuje realizovat zmíněný proces je tzv. “swapping“ domén, během kterého jsou zaměňovány shodné strukturní elementy (motivy/celé domény) v rámci několika identických polysacharidů, čímž jsou nahrazeny vazby v rámci jednotlivých molekul vazbami mezi molekulami.



Obrázek 2.2: Znázornění procesu "swapování" domén. Ze dvou identických monomerů zde vzniká záměnou domény (5-6-7) dimer – ilustrace převzata z [4].

Z pohledu molekulární evoluce lze domény chápat jako moduly "genetického programovacího jazyka", tedy určité mobilní elementy, jejichž kombinacemi v rámci proteinů vznikají sekvence komplexnějších molekul. Zmíněná evoluční funkce byla potvrzena nalezením konkrétní rodiny proteinových domén v rámci genomů různých organismů všech tří základních životních forem (tj. archea, bakterie a eukaryota). Dále bylo zjištěno, že určitá doména se může objevovat jako samostatný protein (jednodoménový) u nižších životních forem a současně jako podjednotka komplexnějšího (multidoménového) proteinu u vyšších organismů.

Proteinové domény lze kategorizovat podle několika atributů. Základním dělením může být následující:

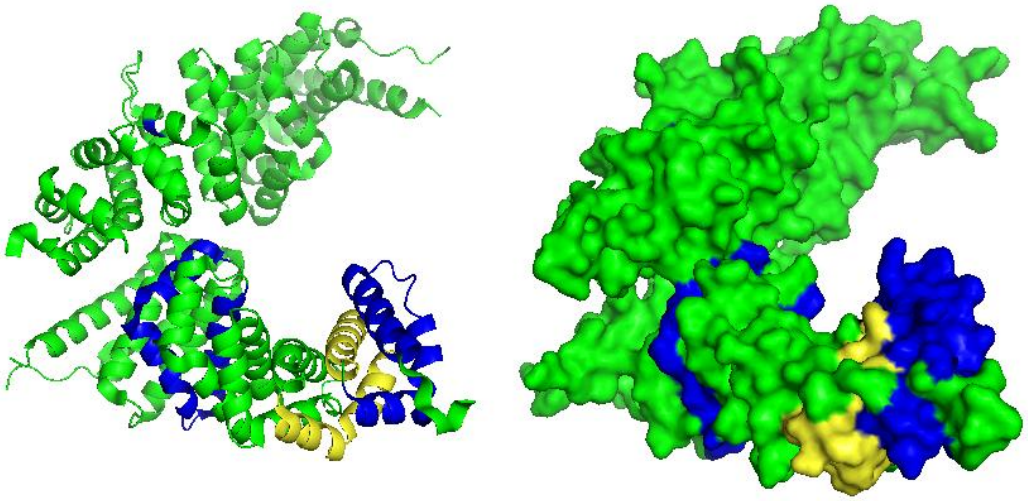
- *Spojité* - tento typ domény je v sekvenci proteinu kódován jako nepřerušená oblast, její automatická detekce je méně problematická. Dobrou zprávou je, že ze všech známých doménových struktur připadá větší část (přibližně 75%) právě tomuto typu.
- *Nespojité* - jejich sekvence je v rámci proteinu přerušena. Tento jev vzniká v multidoménových proteinech během evoluce např. vložením domény/motivu do sekvence jiné domény. Zmíněný typ proto představuje hlavní problém pro metody predikce a především pro ty založené na informaci o sekvenci proteinu.

Další rozdělení je založeno na charakteru elementů sekundární struktury utvářejících konkrétní doménu:

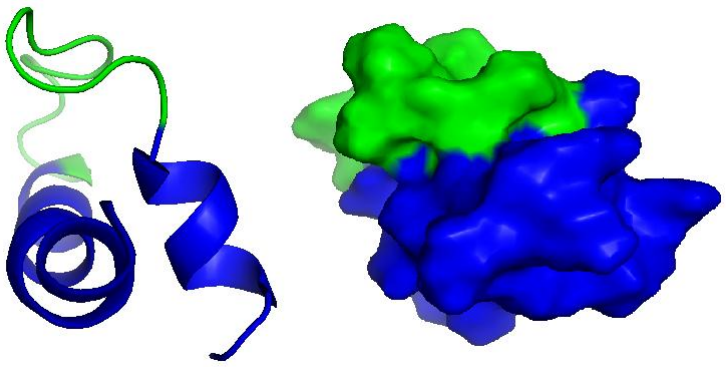
- *All α* - jádro tohoto typu domén je tvořeno výhradně α – spirálami.
- *All β* - jádro tohoto typu je tvořeno výhradně antiparalelními (obvykle dvěma) β - listy
- *α + β* - tato třída obsahuje oba sekundární motivy, bohužel se vlivem překryvu s ostatními špatně rozlišuje a nevyužívá se např. v databázi SCOP
- *α/ β* - doména je vytvořená kombinací motivů β-α-β (TIM soudek) uspořádaných do vrstev

Příklady struktur domén

Na závěr této kapitoly jsem vybral několik běžných proteinových domén, na kterých bych chtěl ilustrovat, jak komplexní struktury mohou nabývat. Domény byly nalezeny v [5], a modely jejich struktury (resp. s nimi spojené proteiny) byly vyhledány v databázi PDB, staženy a následně vizualizovány v programu Pymol. Informace o jejich vlastnostech jsem čerpal z informací v databázi PDB a wikipedie.



Obrázek 2.4: Ilustrace domény Armadillova repetice. Doména byla nalezena v proteinu s PDB kódem 3NMW, konkrétně se jedná o lidský protein obsahující tři výskyty této domény (2x znázorněno modře, jednou žlutě). Doména je tvořena 40 aminokyselinovými residui. Pro tento motiv je typické, že obvykle dochází k jeho několikerému opakování, ze kterého vzniká vlásenková struktura. Levá část obrázku znázorňuje model struktury proteinu se znázorněnými doménami (zobrazení cartoon), prává zobrazuje kompletní 3D strukturu (v zobrazení surface).



Obrázek 2.5: Ilustrace domény Zinkový prst. Doména byla nalezena v proteinu s PDB kódem 1N5G, konkrétně na C-konci lidské DNA polymerázy. Doména je tvořena 38 aminokyselinovými residui. Tento strukturní motiv umožňuje vazbu bílkoviny na DNA (je součástí tzv. transkripčních faktorů). Levá část obrázku znázorňuje model struktury domény (zobrazení cartoon), prává znázorňuje kompletní 3D strukturu (v zobrazení surface).

3 Nástroje pro predikci proteinových domén.

V této kapitole předvedu některé dostupné nástroje pro predikci proteinových domén, popř. nástroje, které jsou s jejich predikcí přímo spojeny. Mezi ně se řadí jak databáze doménových struktur, tak poměrně specificky zaměřené programy. Obecným znakem obou kategorií je, že jsou jednotliví zástupci často vystavěni na konkrétním konceptu - metodě, která zde bude pouze zmíněna a poté případně rozvedena v následující kapitole.

Nejprve se pokusím shrnout důvody, proč je podle mého názoru důležité vyvíjet automatizované metody pro predikci proteinových domén. Základním motivem je využití domén jako prostředku pro dekompozici komplexní trojrozměrné struktury proteinu na jednodušší podjednotky, které následně umožňují její snazší pochopení, systematičtější klasifikaci a také efektivnější porovnávání struktury různých molekul. Další užitečná vlastnost domén plyne z jejich možnou spjatostí s určitou chemicko-biologickou funkcionalitou, ze které lze dále vyvodit určité předpoklady o funkci zkoumaných proteinových molekul. Jedním z nejdůležitějších důvodů, proč je třeba dané metody rozvíjet, je obrovský nárůst objemu informací v oblasti bioinformatiky – kdy se manuální nebo semi-manuální postupy anotace proteinů postupně stávají výkonnostně nedostatečné.

3.1 Databáze proteinových domén

Databáze tvoří poměrně důležitou skupinu nástrojů, které ke svému sestavení využívají metody predikce proteinových domén. V následující části bych se chtěl zaměřit na dva vybrané zástupce, kteří se podílejí na anotacích proteinů v rámci databáze PDB.

3.1.1 Structural Classification of Proteins (SCOP)

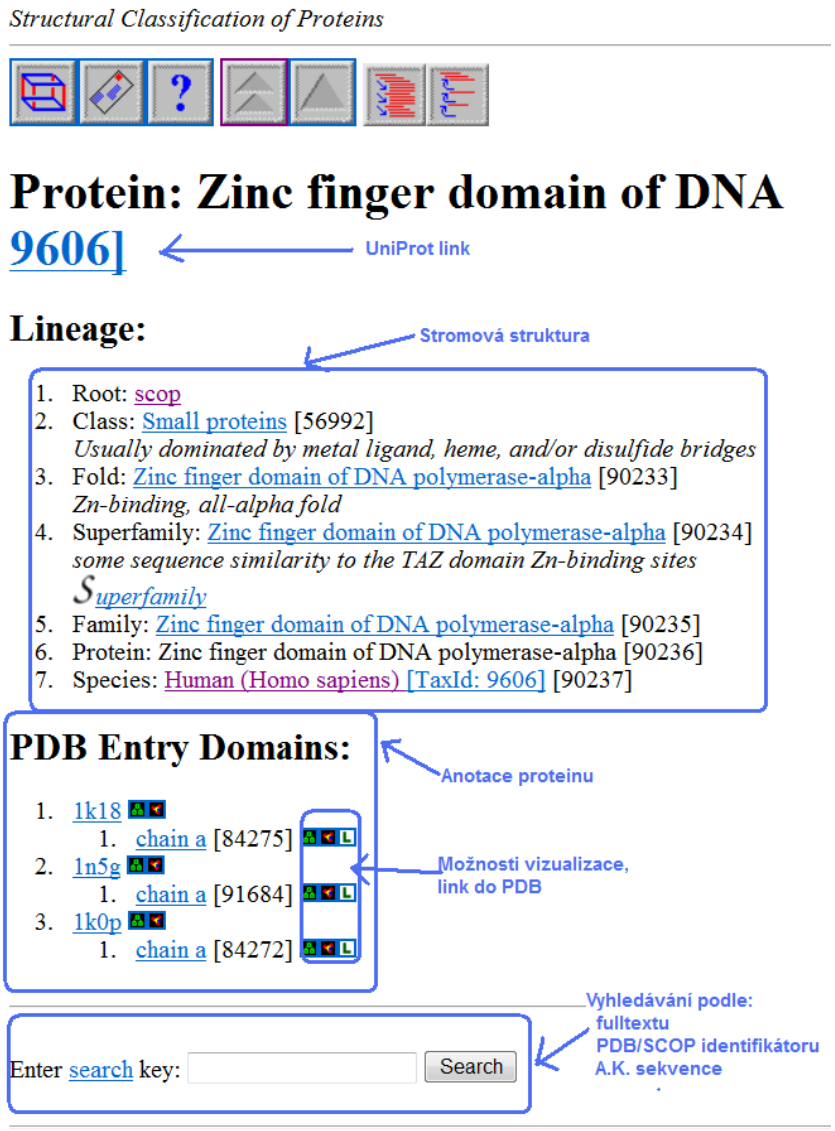
Databáze SCOP je jedna z prvních databází umožňující klasifikovat proteinové molekuly podle struktury a jejich evolučního vývoje. Jako nástroj je silně provázána s databází PDB, která pro ni tvoří zdroj informací o struktuře proteinů (údajný poměr dat v PDB a jejich anotací v SCOP by měl být 1:1). Data mají formu stromové struktury, která zachycuje evoluční a konformační spřízněnost jednotlivých proteinů. Rozhraní pro práci s touto databází je zpracováno formou webu dostupného z adresy: http://scop.mrc-lmb.cam.ac.uk/scop/ nebo pomocí odkazů v rámci anotací jednotlivých proteinů v PDB. Veškeré informace k této části jsem čerpal z [6, 7].

Základní jednotkou klasifikace v rámci této databáze je obvykle proteinová doména. Určování přítomnosti domén v proteinech probíhá dle dostupných materiálů, pomocí ruční vizuální inspekce a porovnání struktury s využitím některých automatizovaných nástrojů. Cílem je dosáhnout použitelné míry škálovatelnosti a všeobecnosti s přihlédnutím k nedostatkům současných plně automatizovaných metod. K hierarchické klasifikaci pak dochází v následujících úrovních:

- Rodina (Family), proteiny jsou do ní zařazovány pomocí metod shlukování na základě dvou kritérií, která by měla zaručovat jejich společný evoluční počátek.
    - Sekvenční identita proteinů je větší než 30%.
    - Sekvenční identita je sice nižší, ale je zřejmá strukturní a funkční podobnost molekul.
- Nadrodina (Superfamily) vznikající spojením rodin, jejichž strukturní a/nebo funkční podobnost vytváří předpoklad, že mají společný evoluční počátek.
- Obvyklé sbalení (Common fold) je vlastností rodin nebo nad-rodin, jejichž hlavní prvky sekundární struktury mají shodné uspořádání se stejným topologickým propojením.
- Třída (class) byla zavedena pro pohodlnější orientaci ve stromové struktuře. Vzniká seskupením rozdílných sbalení (folds) na základě prvků jejich sekundární struktury. Vzniklá hierarchie této kategorie přesně kopíruje rozdělení proteinových domén popsané v předchozí kapitole (all-α, all-β, α+β, α/β, multidoménové proteiny).

Význam databáze, z pohledu této práce, spočívá v tom, že poskytuje významnou množinu dat, na kterých lze trénovat a především testovat automatické metody predikce. K tomuto účelu je v rámci projektu SCOP dokonce připraven soubor dat – vytvořený shlukováním proteinů podle sekvenční podobnosti (s využití metody ASTRAL). Vznikají tak neredundantní databáze PDB_{40}, PDB_{90} apod., kde číslo určuje maximální míru identity sekvencí vybraných proteinů.

Na závěr bych chtěl předvést uživatelské rozhraní, které nabízí možnost vyhledat struktury na základě sekvenční podobnosti (přitom jsou využity různé metody včetně PSI-Blast) a identifikátoru PDB. Další funkcí je podpora procházení stromové struktury (od kořene přes úrovně class, fold, superfamily, family, samotné domény – až po výběr proteinu u příslušného živočišného druhu) a zobrazení vybrané struktury včetně její trojrozměrné vizualizace pomocí nástroje Rasmol.



Obrázek 3.1: Uživatelské rozhraní databáze SCOP. Výstup pro vyhledanou doménu zinkový prst v lidské DNA polymeráze. Jednotlivé složky výstupu jsou popsány v obr.

3.1.2 Pfam

Databáze Pfam byla vytvořena, aby poskytovala soubor konzervovaných proteinových rodin (v tomto případě se rodina rovná doméně), na jejichž základě lze provádět anotace - klasifikace proteinů.
U každé rodině je obsažen kromě možností její vizualizace (s využitím nástroje Rasmol) také krátký textový popis a odkazy na spojené články a publikace. Databáze má na rozdíl od předchozí interní charakter souboru (tzv. flat file), obsahujícího knihovnu profilových skrytých Markovových modelů (HMM), umožňujících vyhledání (resp. sekvenční zarovnání) zástupců jednotlivých rodin. Jako zdroj informací využívá databáze SWISS-PROT a TrEMBL.

Pokrytí sekvencí zmíněných databází anotacemi v Pfam je podle dostupných zdrojů kolem 63%. Informace k této části jsem čerpal z [8, 9].

Základním anotačním prvkem databáze Pfam je, jak bylo řečeno, rodina (resp. doména). Rodiny jsou podle dostupných informací interně spojovány do dvou nepřekrývajících se souborů.

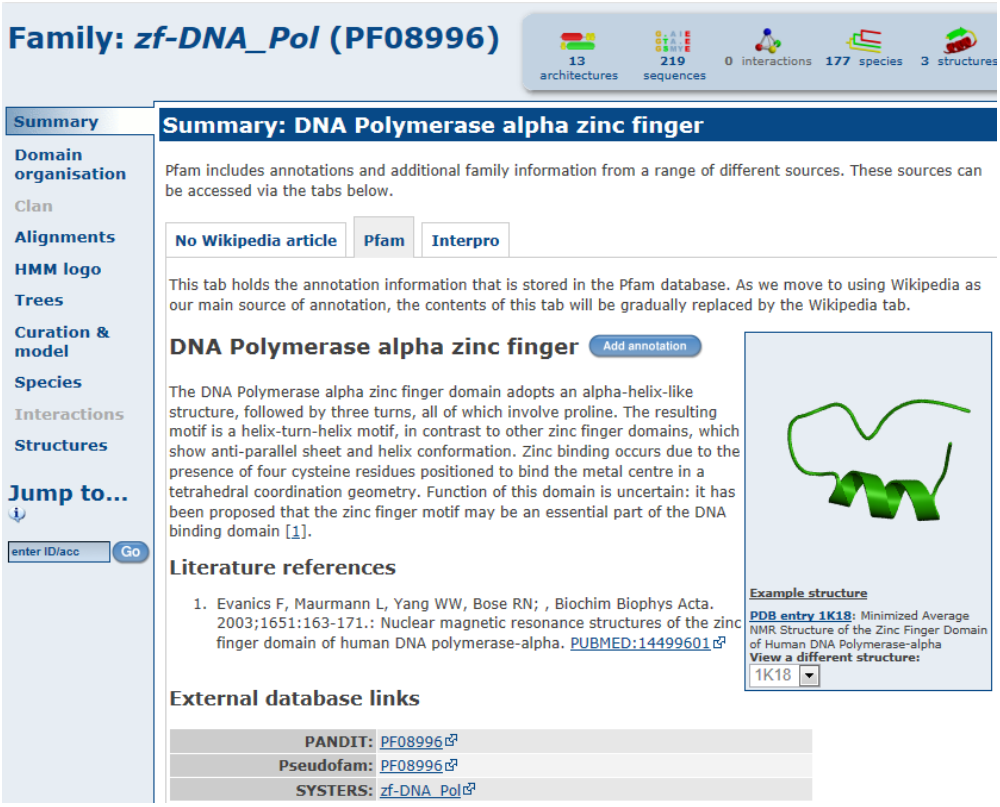
- Pfam-A, základní množina rodin upřednostňovaných pro vyhledávání domén. Celý tento soubor je vytvářen na základě ruční inspekce známých proteinových struktur a každá rodina je v rámci něho reprezentována dvojící zarovnání:
    - Seed - zarovnání několika vybraných sekvencí proteinů reprezentujících danou rodinu (konkrétně těch, které náleží do stejné rodiny po dobu několika obnovení databáze). Každý tento "seed" je reprezentován skrytým Markovovým modelem, umožňujícím vyhledat další reprezentanty dané domény. Zmíněný skrytý Markovův model je generován algoritmem HMMER (v současné době ve verzi 3). Hlavním účelem tohoto zarovnání je pak podpora škálovatelnosti v rámci operací nad celou databází - především vyhledávání.
    - Plné zarovnání – vzniká nalezením tzv. homologů ve vstupních datech (během procesu obnovy databáze) s využitím vytvořených skrytých Markovových modelů. Nalezení homologové jsou následně filtrováni na základě určené prahové hodnoty a poté zarovnáni s tzv. profilem.
- Pfam-B, tento soubor rodin byl vytvořen jako rozšíření původního konceptu Pfam. V aktuálních verzích databáze lze tuto část také využít k anotaci nových proteinů. Vyhledávání v rámci této množiny probíhá mírně odlišně od předchozí (založené na zarovnáních typu seed a aplikací HMM). V prvním kroku jsou pomocí algoritmu BLAST2 nalezeny odpovídající segmenty reprezentující určitou proteinovou rodinu, následně je za běhu sestaven profilový HMM a s jeho pomocí je vstupní sekvence zarovnána vzhledem k nalezené rodině. Poměrně zajímavou vlastností souboru – z pohledu této práce je, že vzniká s využitím databáze Domainer (která vzniká automatickou predikcí proteinových domén (metodou mkdom2) nad daty ze stejných vstupních databází jako Pfam).

Kromě zmíněných možností anotace a klasifikace proteinů nabízí databáze širší možnosti uplatnění. S použitím rozšíření NIFAS lze například vytvářet evoluční strom struktury konkrétní domény, strom je vytvářen metodou spojování sousedů realizovanou algoritmem FastTree nad oběma typy zarovnání rodin (úplné i seed). Dalším zajímavým prvkem, který bych chtěl dále rozvést v implementační sekci následující práce, jsou informace spojené přímo s HMM, a to sekvenčního loga jednotlivých rodin (domén).

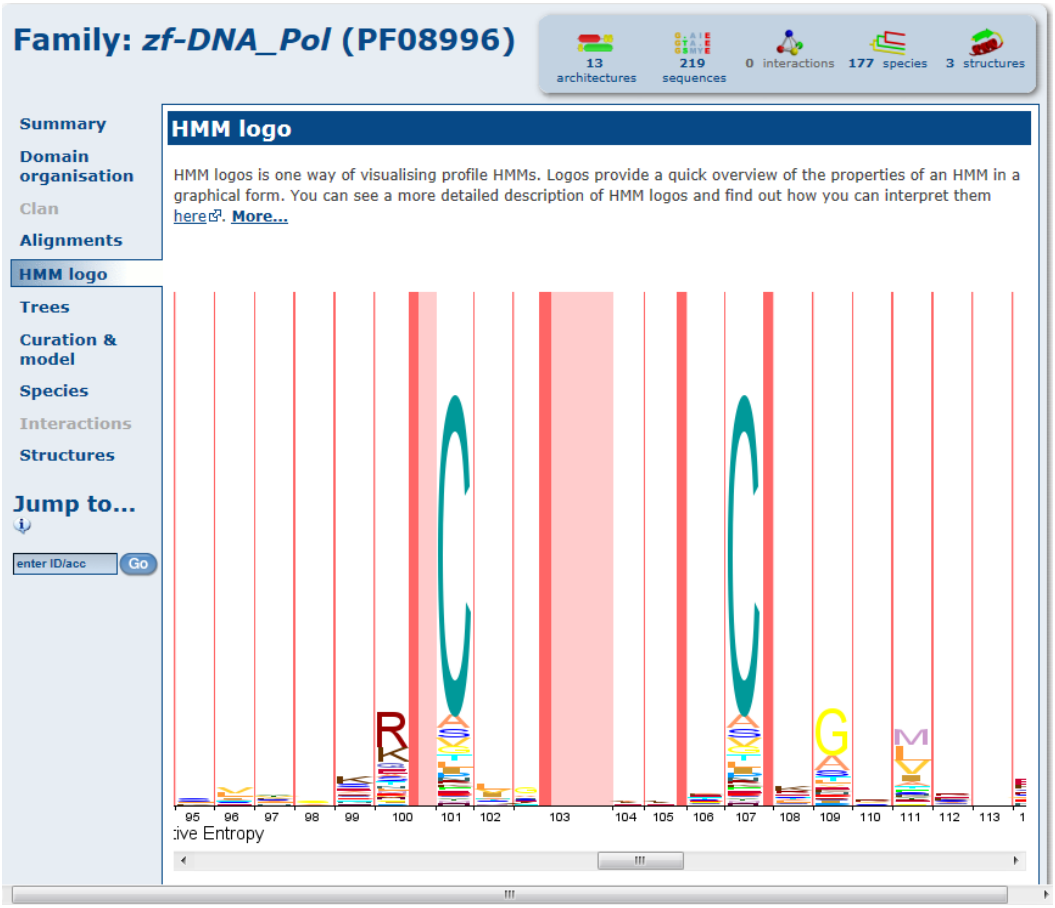
Uživatelské rozhraní pro práci s databází je dostupné na trojici webových serverů, které obsahují shodnou základní funkcionalitu – vyhledávání. V rámci specifických rozšíření se ale jednotlivé implementace liší. Pro přístup z různých částí světa jsou dostupné tyto servery:

- Velká Británie: http://pfam.sanger.ac.uk/
- USA: http://pfam.janelia.org/
- Švédsko: http://pfam.sbc.su.se/

Na závěr bych chtěl uvést několik detailů k uživatelskému rozhraní Pfam (běžícímu na britském serveru). V rozhraní můžeme získat anotaci (z pohledu domén) vložené sekvence proteinu, je zde obsaženo fulltextové vyhledávání, vyhledávání domén podle identifikátoru rodiny, nebo PDB identifikátoru.



Obrázek 3.2: Uživatelské rozhraní databáze Pfam (server pro Velkou Británii) poskytující - základní informace o doméně zinkový prst. Je zde zastoupena možnost vizualizace domény a z dalších voleb bych upozornil na záložku trees – poskytující evoluční strom.



Obrázek 3.3: Ve shodném uživatelském rozhraní databáze Pfam byla přechodem na záložku HMM logo získána vizualizace sekvenčního loga domény zinkový prst.

3.2 Webové servery pro predikci proteinových domén

V této části se pokusím prozkoumat, několik nástrojů pro analýzu proteinových sekvencí, které jsou volně dostupné ve formě webového serveru. Zmíním použité metody a některé z nich pak rozvedu v příslušné kapitole.

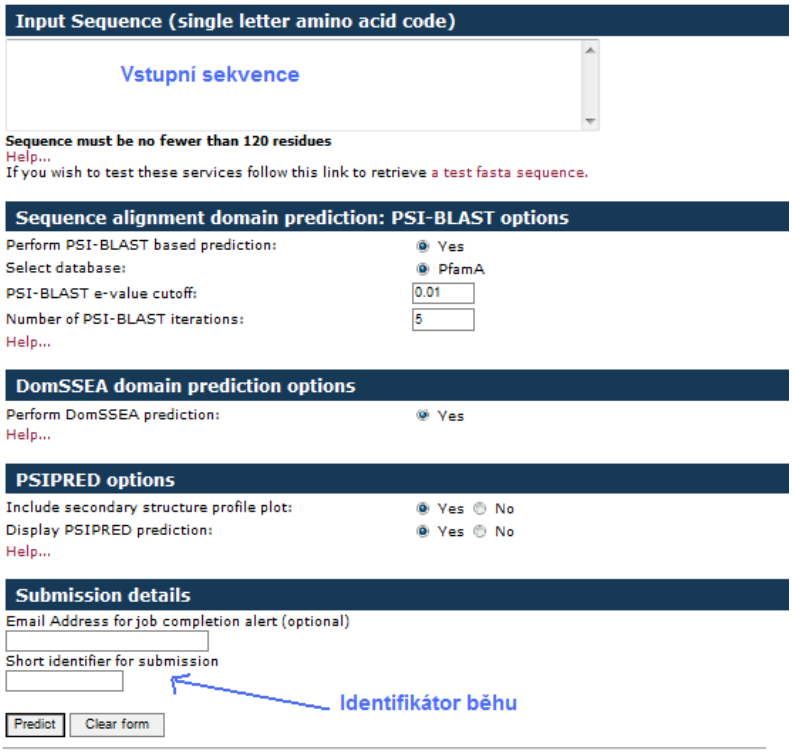
3.2.1 DomPred (Preotein Domain Prediction Server)

Tento server je provozován londýnskou univerzitou (University College London) a pracuje na principu predikce proteinových domén založené na informaci o sekvenci aminokyselin. Základní informace o fungování nástroje jsem získal z [10]. Uživatelské rozhraní je dostupné z adresy: http://bioinf.cs.ucl.ac.uk/dompred.

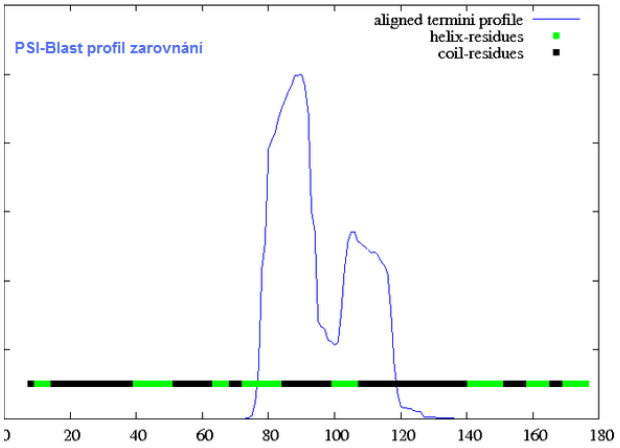
Implementace serveru využívá podle dostupných informací k predikci domén trojici komponent, které jsou postupně využívány v závislosti na výsledcích predikce jednotlivých částí.

- Pfam-A vyhledávání slouží jako primární prostředek analýzy vstupní sekvence. Pracuje na principu lokálního - globálního zarovnání vstupu proti sekvenci domény nalezené v databázi Pfam (tato je popsána detailněji v předchozí sekci).
- PSI-Blast a DomSSEA, tato dvojice algoritmů je využita, pokud databáze nevrátí žádný dostatečně věrohodný výsledek. PSI-Blast vytváří tzv. profil (PSSM) s využitím metod lokálního zarovnání a následného prohledání interní databáze. Profil je následně využit metodou DomSSEA – tato metoda pracuje na principu ab-inito. Konkrétně domény předpovídá zarovnáním predikované sekundární struktury (algoritmus PSIPRED) ke známým šablonám [11].

Uživatelské rozhraní serveru umožňuje vkládat dotazovou sekvenci aminokyselin ve formátu FASTA (resp. pouze sekvenci aminokyselin). Dále je možné provést nastavení různých parametrů metod použitých k predikci. Pro spuštění procesu predikce je třeba zadat libovolný krátký identifikátor, který je použit k identifikaci výsledků běhu. Predikce probíhá podle instrukcí po dobu 30-120 minut. (Vyzkoušeno na části proteinu s identifikátorem 1H1O, která obsahuje dvě domény a 183 aminokyselin. Doba běhu byla 12 minut). Výsledky běhu server uchovává pod zadaným identifikátorem a uživatel může být informován o dokončení jeho úlohy pomocí e-mailu.



Obrázek 3.4: Rozhraní serveru DomPred – umožňující zadat novou úlohu predikce proteinových domén v zadané sekvenci.



Putative Domain boundaries found by PSI-BLAST:

Putative domain boundaries located in PSI-BLAST alignment profile:
Number of predicted domains by DPS: 2
Domain Boundary locations predicted DPS: 90
Number of PSIBLAST hits = 7008
Download PSI-BLAST output
Download PSI-BLAST data table

DomSSEA Results

Score	Match	No. Doms	Boundaries	SCOP code
0.90756303	1fcdD	2	99	a.3.1.4, a.3.1.4
0.85714287	1fcdC	2	99	a.3.1.4, a.3.1.4
0.8338028	1h1oA	2	99	a.3.1.4, a.3.1.4
0.82719547	1h1oB	2	99	a.3.1.4, a.3.1.4
0.7989276	1etpA	2	72	a.3.1.4, a.3.1.4
0.7935657	1m6zC	2	72	a.3.1.4, a.3.1.4
0.7882038	1m6zD	2	72	a.3.1.4, a.3.1.4
0.7566265	1sh5B	2	99	a.40.1.1, a.40.1.1
0.75354105	1oxjA	2	85	a.60.1.2, a.118.1.13

Obrázek 3.5: Části uživatelského rozhraní serveru DomPred , zobrazující výsledky (úspěšné) predikce. V horní části je profil zarovnání metodou PSI-Blast a ve spodní části je zobrazen výsledek predikce metodou DomSSEA – obsahující predikovaný počet a hranici domén a jím příslušný identifikátor databáze SCOP.

3.2.2 Myhits – MkDom2

Myhits je veřejně dostupná databáze zaměřená na proteinové domény. Na tomto místě ji uvádím, protože kromě doménové databáze nabízí tento projekt také webový server s množství analytických nástrojů, mezi kterými se nachází implementace metody MkDom2 pro predikci proteinových domén. Tato metoda byla zmíněna v předchozí kapitole a je využita k automatické anotaci proteinů, kterou vzniká databáze Domainer. Bohužel k funkci tohoto serveru jsem nenašel podrobnější dokumentaci. Webové uživatelské rozhraní metody Mkdom2 je dostupné z adresy: http://myhits.isb-sib.ch/cgi-bin/mkdom2.

3.2.3 InterProScan

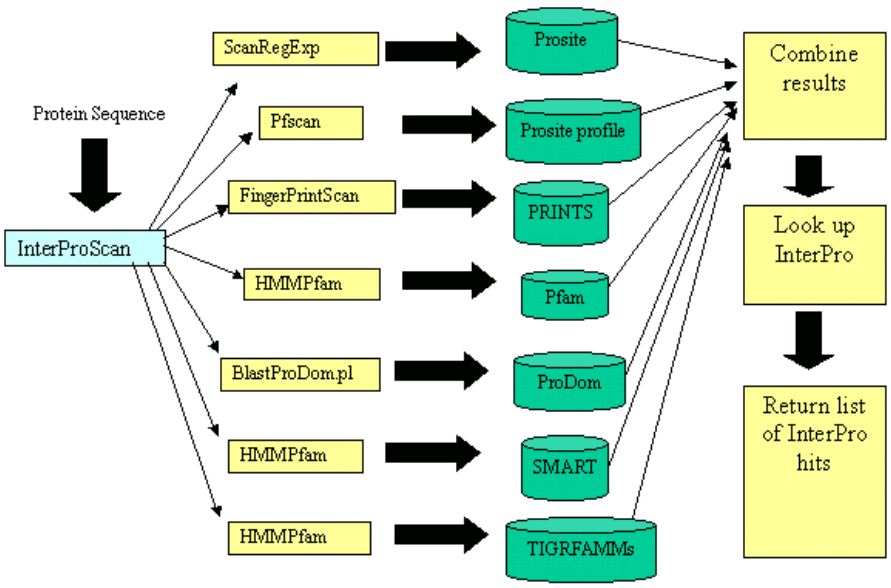
Je další typ utility zpracované ve formě webového serveru. Metoda využívá kombinace několika přístupů k analýze vložené sekvence. Dotaz je zpracován proti databázím, které jsou členy InterPro

databáze (InterPro vzniká spojením dílčích anotací všech členských databází pro konkrétní protein, členy toho konsorcia jsou databáze Pfam, ProDom, Prosite, PRINTS, SMART a TIGRFAMMs). Tímto přístupem se metoda snaží redukovat nevýhody jednotlivých dílčích zástupců. Server je provozován Evropským bioinformatickým institutem (EMBL-EBI) a webové uživatelské rozhraní je dostupné z adresy: http://www.ebi.ac.uk/Tools/pfa/iprscan/. Informace o metodě jsem čerpal z [12].

Činnost tohoto nástroje začíná vložením sekvence v jednom z podporovaných formátů (GCG, FASTA, EMBL, PIR, NBRF, UniProtKB/Swiss-Prot format). Následně je spuštěna řada nástrojů, které paralelně prohledávají databáze skupiny InterPro:

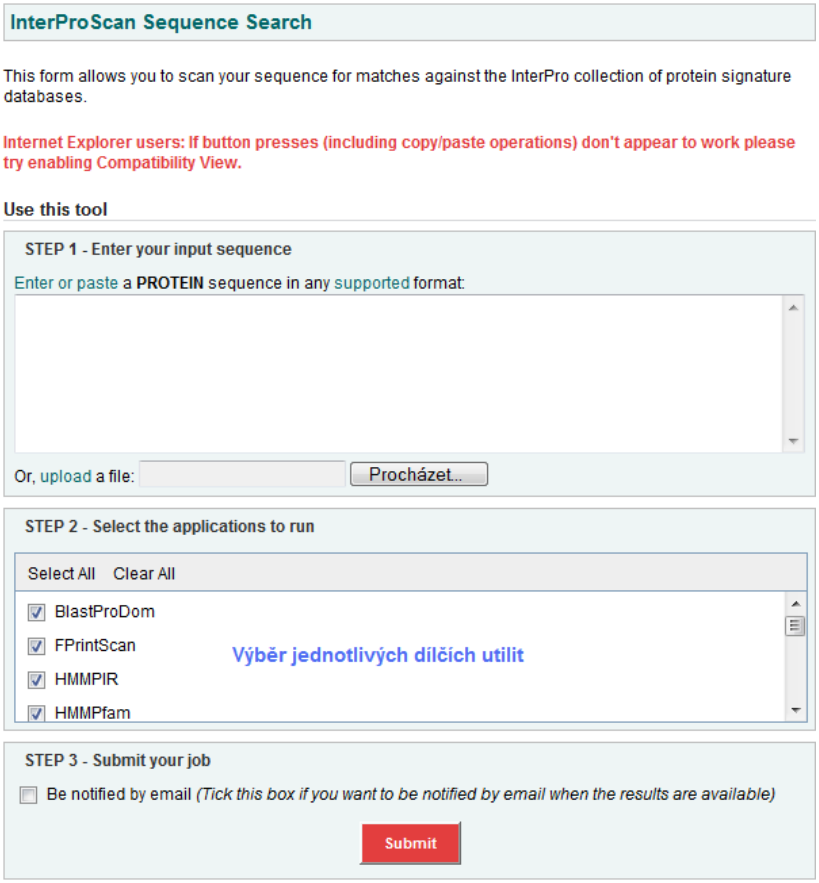
- ScanRegExp prohledává regulární výrazy v databázi PROSITE,
- PfScan prohledává profily v rámci databáze PROSITE,
- FingerPRINTScan prozkoumává tzv. "fingerprints" (skupiny motivů, vytvářené na základě biologického kontextu) v databázi PRINTS,
- HMMPfam vyhledává ve skrytých Markovových modelech v databázích Pfam, TIGRFAMMs a SMART,
- BLASTProDom.pl vyhledává rodiny v ProDom databázi.

Každá z jednotlivých komponent vrací seznam nalezených struktur, tyto seznamy jsou zkombinovány a jako výsledek jsou vráceny rodiny z InterPro, které jim přísluší.

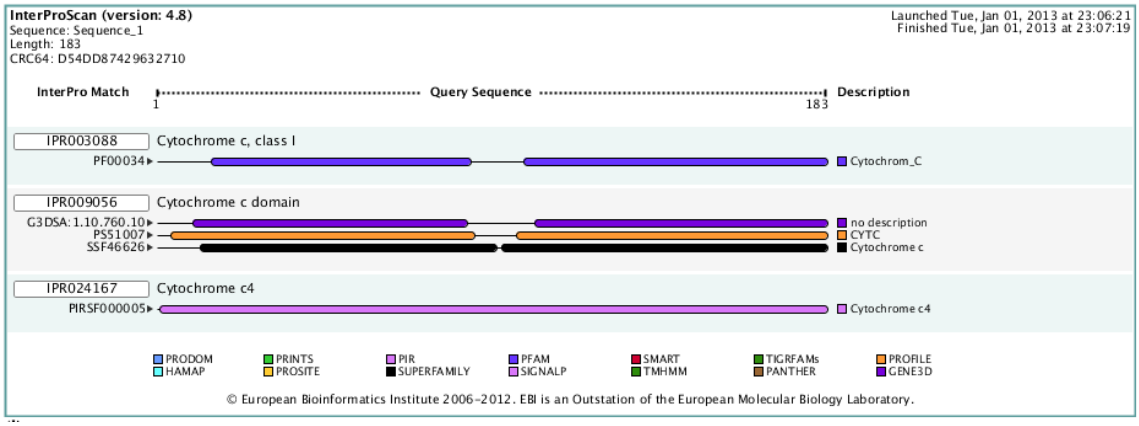


Obrázek 3.6: Schéma fungování metody InterProScan převzaté z [13]. Na obrázku jsou znázorněny jednotlivé dílčí nástroje a databáze, proti kterým se dotazují.

Rozhraní metody nabízí možnost vložit sekvenci aminokyselin v podporovaném formátu k analýze a lze vybrat jednotlivé komponenty použité k interakci s databázemi v rámci InterPro. Opět lze vložit email a identifikátor k informování uživatele o výsledku běhu jeho analýzy, který je serverem uchován. Metoda byla opět testována na shodné sekvenci jako v případě DomPred (část proteinu 1H1O, predikce dopadla úspěšně a v porovnání s předchozí metodou v kratším čase, a to za 3 minuty). Výsledek lze zobrazit ve vizuální a v souborné tabulární formě.



Obrázek 3.7: Webové rozhraní metody InterProScan

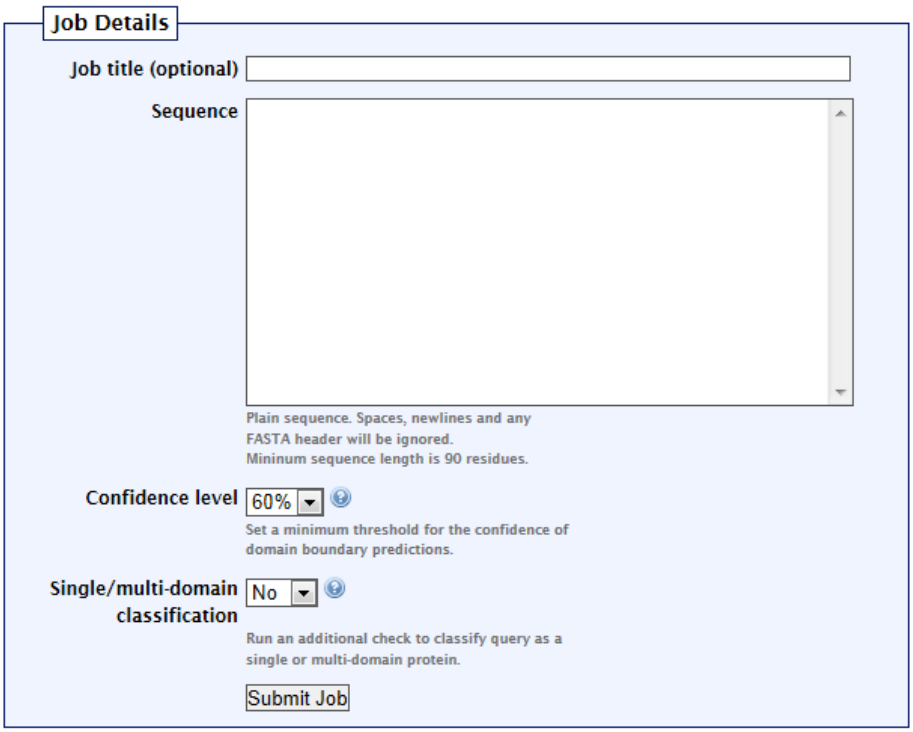


Obrázek 3.8: Výstup metody InterProScan pro část proteinu s identifikátorem 1P1O obsahující 2 domény.

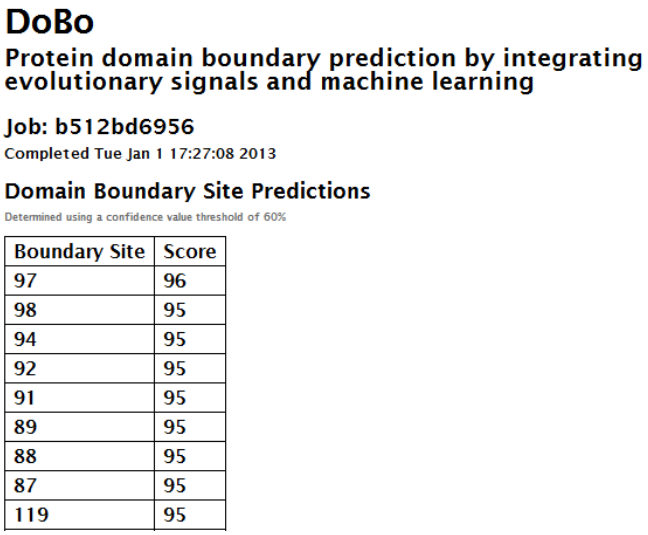
3.2.4 DoBo

Následující webový server tvoří rozhraní stejnojmenné metody pro predikci proteinových domén, pracující na principu strojového učení. Metoda rozpoznává hranice domén ze sekvence aminokyselin s použitím metody PSI-Blast (pracující s neredundantní databází - NR) a dvojí aplikace klasifikátoru SVM. Metodu více prozkoumám v příslušné kapitole. Informace k této části jsem čerpal z [14], uživatelské rozhraní metody je dostupné z: http://sysbio.rnet.missouri.edu/dobo/. Server je provozován Univerzitou v Missouri.

Rozhraní webového serveru nabízí obdobné možnosti jako ostatní nástroje, analyzovanou sekvenci lze vložit v textové podobě. Konfigurovat můžeme úroveň spolehlivosti (procento predikovaných hranic domén, které by měly náležet do rozmezí +/- 20 residuí od reálné hranice) a aktivaci dodatečného testu, který kontroluje, zda jde o multidoménový protein. Server jsem testoval opět na shodné části proteinu s identifikátorem 1P1O jako v ostatních případech. Hranice domény byla predikována na správné pozici v čase 12 minut. Výsledky aplikace prezentuje v jednoduché tabulkové formě.



Obrázek 3.9: Základní rozhraní metody DoBo



DoBo

Protein domain boundary prediction by integrating evolutionary signals and machine learning

Job: b512bd6956

Completed Tue Jan 1 17:27:08 2013

Domain Boundary Site Predictions

Determined using a confidence value threshold of 60%

Boundary Site	Score
97	96
98	95
94	95
92	95
91	95
89	95
88	95
87	95
119	95

Obrázek 3.10: Část uživatelského rozhraní metody DoBo zobrazující výsledek predikce proteinových domén

3.2.5 Zhodnocení kapitoly

Na závěr této kapitoly bych chtěl poznamenat, že kromě zde popsaných nástrojů existuje celá řada dalších zástupců. Mnohé z nich jsou implementovány formou konzolové aplikace nebo aplikačního serveru a jsou převážně volně dostupné. Principy použité u některých z nich prozkoumám v následující kapitole, která se zabývá používanými metodami.

4 Metody predikce proteinových domén

V následující kapitole práce chci navázat na část předchozí a hlouběji se zaměřit na existující metody predikce, prozkoumat přístupy, kterých využívají a pokusit se shrnout přínosy a nedostatky jednotlivých zástupců. Informace prezentované v této kapitole jsem čerpal především z [11].

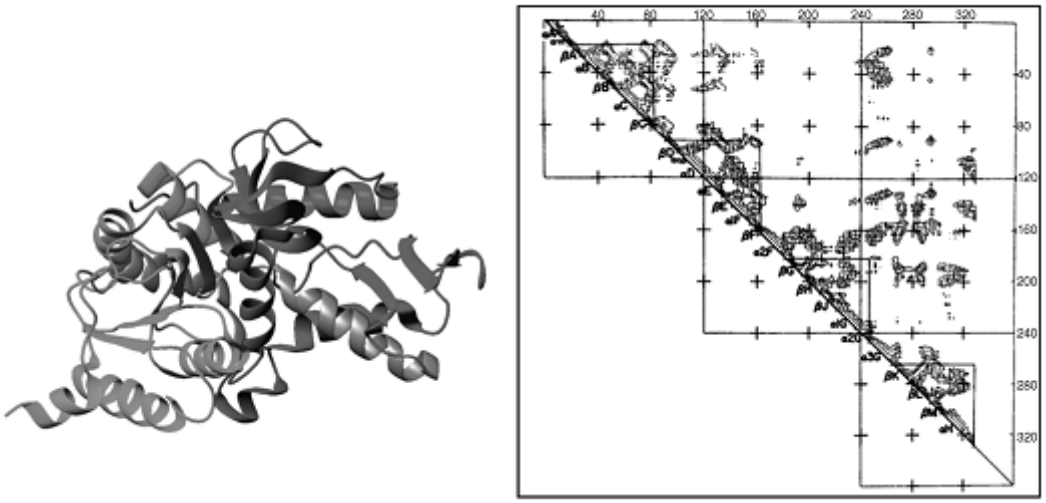
Aktuálně existuje poměrně široké spektrum algoritmů umožňujících s různou mírou úspěšnosti predikovat proteinové domény. Většina těchto postupů se dá podle typu informace, se kterou pracují, zařadit do jedné ze dvou základních skupin. První skupinu představují metody využívající informaci o vnitřní struktuře proteinové molekuly, velmi často charakterizované pomocí polohy alfa uhlíku jednotlivých aminokyselin. Druhá skupina je tvořena vesměs novějšími postupy, které naopak vycházejí z informace o sekvenci aminokyselin v konkrétním proteinu. Obě skupiny metod budou dále popsány v následujícím textu. Kromě těchto dvou základních tříd, existuje ještě celá řada algoritmů využívajících různými způsoby kombinace obou předchozích přístupů, tyto budou okrajově popsány v závěru kapitoly.

Značný počet různých metod predikce pouze reflektuje zájem, jakému byla tato oblast výzkumu proteinových struktur v posledních přibližně třiceti letech vystavena. I přes poměrně enormní snahu stále ještě existuje celá řada problémů, které se nepodařilo uspokojivě vyřešit. Pro plně automatizované postupy je navíc často velmi komplikované vyrovnat se s poznatky, které přináší ruční analýza nově získaných proteinů, kde se mohou vyskytovat taková rozdělení struktury, která jsou v rozporu s některými z předchozích případů. Nelze proto zatím považovat nějaký postup za dostatečně spolehlivý a provádět s ním plně automatizovanou predikci u nových molekul, což je například celkem běžné u predikce sekundární struktury. Na druhou stranu přináší výpočetní metody jednu důležitou výhodu oproti metodám expertním - jejich výstup je rychlý a konzistentní.

V [11] jsou dále zmíněny některé dílčí problémy, které postihují oblast jako celek. Jedním z hlavních je, že dlouho dobu neexistovala sjednocující metodika, která by umožnila porovnávat výsledky jednotlivých algoritmů. Kromě toho většina autorů využívala ručně (resp. odborníky z oboru krystalografie) sestavenou testovací sadu dat, s různou distribucí známých struktur. V současné době se od tohoto přístupu ustupuje a využívají se sady dat založené na shodě mezi databázemi SCOP, resp. CATH. Pokud bude dále v této kapitole zmíněno hodnocení některých zástupců, pak bude vycházet z [11], kde autor využívá k testování metod sadu dat sestavenou na základě shody struktur mezi zmíněnými databázemi, která navíc obsahuje každý typ kombinace topologií popsaných v CATH na klasifikační úrovni topologie.

4.1 Metody založené na vnitřní struktuře molekul

Algoritmy využívající informace o vnitřní struktuře proteinu vznikaly dříve než zástupci ostatních skupin. V rané fázi se autoři snažili přímo zautomatizovat postupy použité odborníky v oboru krystalografie při manuální analýze proteinů. Jedním z běžných manuálních přístupů k predikci byla v této době vizuální inspekce grafu vzdáleností alfa uhlíků. Tento postup posloužil jako základ vůbec prvního systematického algoritmu pro vyhledávání domén ve struktuře proteinů (Rossman a Liljas, 1974).



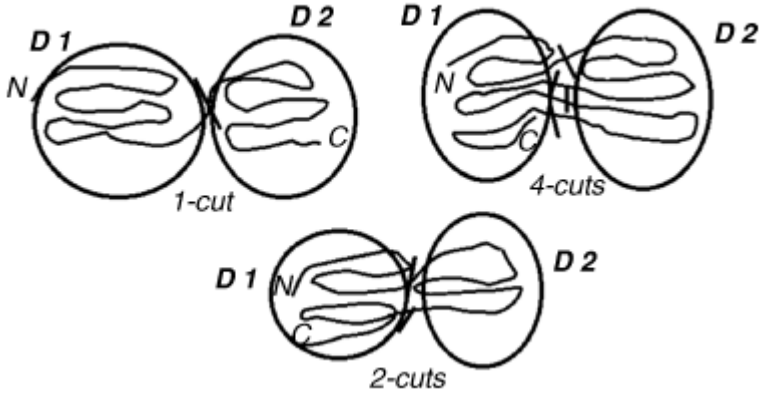
Obrázek 4.1: V levé části je vykreslený model (ribbon) proteinu LDH. V pravé části je mapa vzdáleností C alfa uhlíků sloužící jako vstupní zdroj informací o prostorovém uspořádání proteinu. Jednotky sekundární struktury jsou hledány podél vyznačené diagonály, trojúhelníky v mapě pak vyznačují místa s vyšší hustotou výskytu krátkých vzdáleností mezi Cα. První dva trojúhelníky (zprava) představují pozici NAD-vazebné domény a zbylé dva katalytickou doménu. Převzato z [11].

Předchozí postup, stejně jako princip většiny současných metod, je vystaven na jednoduché úvaze. Základ zmíněné úvahy spočívá v hledání takové skupiny residuí, obsahující vyšší počet atomických kontaktů v rámci této skupiny (resp. domény) než mezi ní a okolím, což přímo vychází ze strukturálních a termodynamických vlastností proteinových domén:

- Stabilita
- Kompaktnost
- Přítomnost hydrofobního jádra
- Schopnost procházet procesem sbalení nezávisle na zbytku proteinu

Ze zmíněných bodů je nutné upozornit na prostorovou kompaktnost domén, jež přináší jednu významnou komplikaci (významnou především pro metody pracující se sekvencemi, které se s ní

často nedokážou vyrovnat) a tou je existence tzv. nesouvislých domén. V rámci struktury zmíněných útvarů dochází k tomu, že v prostoru si blízké oblasti mohou být poměrně vzdálené v rámci řetězce aminokyselin. Pro rozebíranou skupinu algoritmů nepředstavuje tato vlastnost proteinových domén větší problém.



Obrázek 4.2: Příklad převzatý z [11], na kterém je dokumentován problém rozdělení trojrozměrné struktury proteinu mezi domény. Pokud jsou obě domény souvislé, je řetězec rozdělen na jednom místě (1-cut), pokud je jedna doména souvislá a druhá nesouvislá, což vzniká například vložením genu, musí být řetězec rozdělen na dvou místech (2-cut). Pokud jsou obě domény nesouvislé, je nutné rozdělit řetězec na více místech (4-cut). Podobných rozdělení bývá obvykle generováno větší množství s cílem vybrat takové, které nejlépe uspokojuje základní úvahu zmíněnou výše.

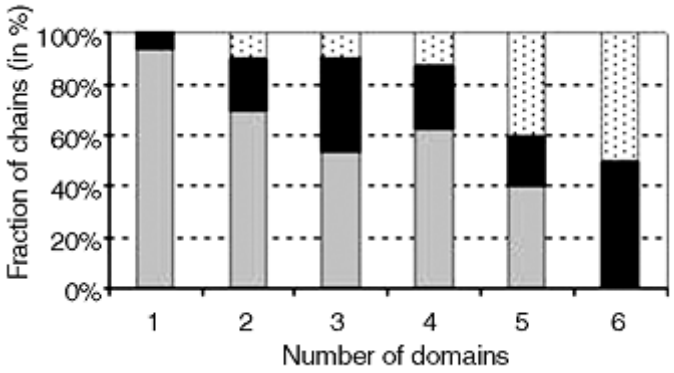
Metody využívající přístup popsaný výše obecně pracují na principu zjednodušeně popsaném na předchozím obrázku, kdy se snaží hierarchicky rozdělovat strukturu proteinu. Toto rozdělení na vyšších úrovních své hierarchie dovoluje lokalizovat přímo proteinové domény a nižších jednodušší motivy (kombinace prvků sekundární struktury).

V následující části textu již budou prezentováni vybraní zástupci algoritmů této skupiny, kde to bude možné, uvedu srovnávací výsledek metody prezentovaný v [11].

4.1.1 PUU (Parser for protein folding unit)

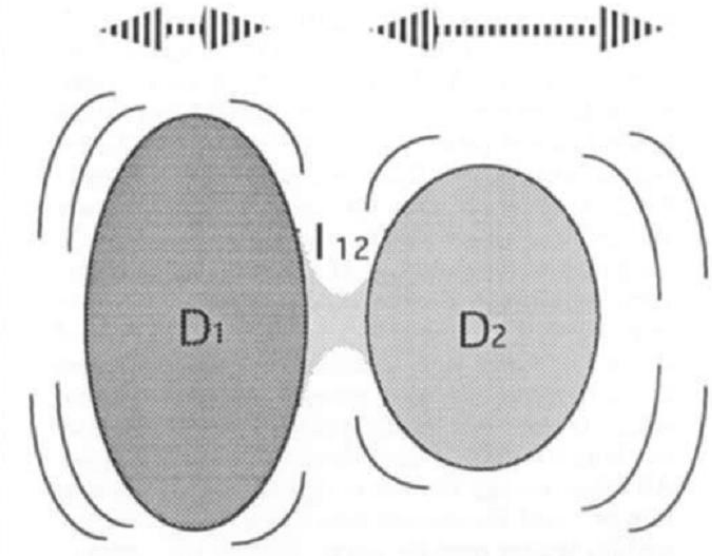
Algoritmus vytvořený autory Chrisem Sanderem a Liisou Holm, popsaný v [11, 15] představuje typického zástupce právě definované třídy, který plně využívá obecné principy popsané výše. Metoda pochází z roku 1994, při určování domén postupuje metodou "od shora dolů" (celá sekvence je postupně hierarchicky rozdělována na menší podjednotky). Výsledky metody jsou testovány a srovnávány s obdobnými metodami (NCBI, PDP a DomainParser) v [11] pomocí speciálně navržené

testovací sady proteinů, která byla zmíněna v předchozím textu. Výsledky metody pro jednotlivé kombinace topologií jsou prezentovány na následujícím obrázku.



Obrázek 4.3: Hodnocení metody převzaté z [11]. Šedá část představuje úspěšnou predikci. Černá část představuje podíl struktur, kde byl predikován větší počet domén, tečkovaná část představuje naopak struktury, u kterých byl predikován nižší počet domén.

Algoritmus je vystaven na principu popsaným Sanderem, který využívá kriteria maximalizace času mezi doménové fluktuace, založeném přibližně na následující úvaze. Protein se během svého rozbalování rozpadá na podjednotky (resp. strukturální domény), které mají kompaktní tvar a jejichž vzájemné interakce jsou slabé. Tato intuitivní definice je pak kvantifikována jednoduchým modelem.



Obrázek 4.4: Model prvotní fáze rozbalení proteinu, která vedla k rozpadu struktury do dvou podjednotek D_1 a D_2, rozhraní mezi jednotkami je označeno jako I_{12}. V rámci rozhraní probíhají interakce obou jednotek pomocí nevazebných atomových interakcí. Převzato z [15].

Na předchozím obrázku je znázorněn zmíněný model pro první fázi rozbalování proteinu, V tomto systému dochází k pomalému relativnímu pohybu vzniklých podjednotek a vzájemnému přeskupování lokální struktury proteinu a přítomného rozpouštědla. Což vede k postupnému vstupu rozpouštědla do oblasti rozhraní obou struktur a vyvrcholí jejich úplnou separací. Aby k podobnému procesu vůbec došlo, musí být relativní pohyb jednotek dostatečně pomalý. K určení výsledné dekompozice proteinu je pak nutné najít takové rozdělení jednotek, pro které je konstanta relativního pohybu (τ) největší. Výpočet konstanty je prezentován rovnicí (1), kde μ představuje snížení hmotnosti a I_{12} sílu rozhraní mezi jednotkami 1 a 2.

$$\tau^2 \sim \frac{\mu}{I_{12}} \tag{1}$$

Konstanta τ je v rámci metody určena na základě kvantitativního odhadu, založeném na počítání atomů a meziatomových kontaktů. Metoda při stanovení odhadu vychází z mapy kontaktů mezi Cα a uvažuje:

- Vodíkové můstky mezi atomy polypeptidové páteře v rámci β-listů
- Methylové skupiny
- Vodíkové vazby
- Van der Waalsovy síly.

Pro každou výše vyjmenovanou skupinu je určena síla vazby (v kcal/mol) a maximální možný počet meziatomových kontaktů. I_{12} se pak vypočte jako suma kontaktů mezi atomy v daném rozhraní a úbytek hmotnosti μ podle vztahu (2), kde m_c je hmotnost atomu uhlíku (12g/mol) a N_1, N_2 jsou počty nevodíkových atomů v rámci obou podjednotek.

$$\mu = m_c[N_1 N_2/(N_1 + N_2)] \tag{2}$$

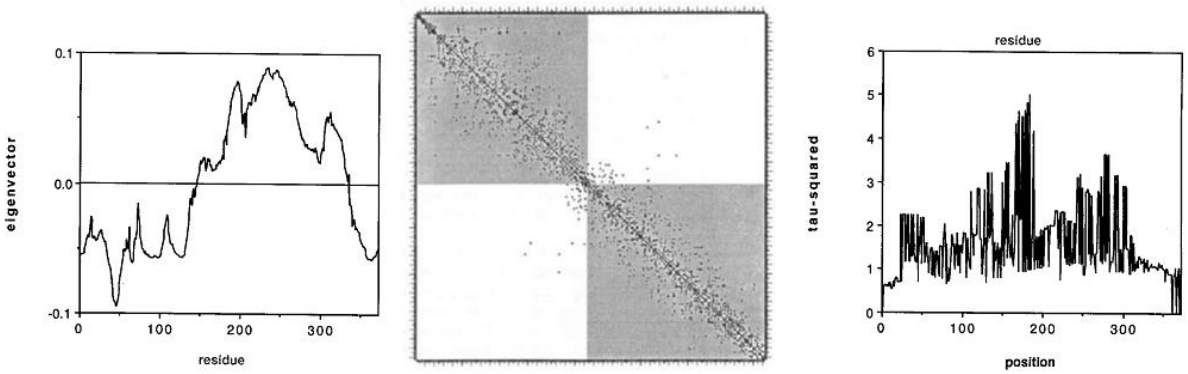
Celý problém se redukuje, jak bylo řečeno výše, na nalezení takového rozdělení jednotek D_1 a D_2, aby byla hodnota $\tau(D_1, D_2)$ maximální. Tento stav koresponduje se situací, kdy jsou řádky/sloupce kontaktní matice uspořádány (permutací) tak, aby řádky/sloupce *1, ..., k* náležely jednotce D_1 a zbylé řádky/sloupce D_2 a současně řez za *k* minimalizoval kontakty mezi oběma jednotkami.

Minimalizace kontaktů řezem *k* mezi hledanými jednotkami je dosaženo pomocí shlukování silně interagujících residuí. Vytvoření shluků je dosaženo specifickou metodou ordinace nazývanou reciproční průměrování [21]. Tato metoda je užívána ve statistice k analýze kontingenčních tabulek, jejím základem je efektivní nalezení vlastních vektorů. V rámci algoritmu je aplikována na problém nalezení vlastních vektorů symetrické pozitivně semi-definitní matice kontaktů *A*. Za tímto účelem

hledá konzistentní soubor vah x_i pro jednotlivá residua. Problém je vyjádřen následujícím vztahem (3), který metoda iterativně řeší vzájemným průměrováním nových vah *x'* a předchozích vah *x*, r_i zde označuje sumu kontaktů v řádku.

$$x_i' = \frac{\sum_j a_{ij} x_j}{r_i}, kde\ r_i = \sum_j a_{ij} \tag{3}$$

Algoritmus jako výsledek výše popsané metody vezme první netriviální vlastní vektor a s jeho pomocí seřadí residua v kontaktní matici. Tímto krokem se redukuje problém hledání N řezů v lineární sekvenci na průchod přeskupené sekvence a vyhledávání vhodného místa dělení (sekvence je rozdělena mezi dvě domény, dochází tedy k bisekci). Jednotlivá kandidátní místa jsou hodnocena podle vztahu (1). Dále jsou filtrována taková rozdělení, která by vedla ke vzdálenosti řezů menší než 10 residuí. Nakonec je vybrán dělicí bod, který maximalizuje vztah (1). Metoda může rekurzivně dělit nově vzniklé části, pokud je jejich velikost větší než 80 residuí (z důvodu dodržení pravidla globularity).



Obrázek 4.5: Levý graf představuje nalezený vlastní vektor vynesený proti pozicím residuí. Vertikální čára představuje kandidátní bod dělení (residua pod resp. nad linií náleží stejné doméně). Prostřední diagram znázorňuje kontaktní matici uspořádanou dle nalezeného vektoru, šedé části označují bloky, které jsou rozděleny jediným bodem dělení. Na pravém grafu je vykreslen profil τ^2 nad ordinací uspořádanou sekvencí. Graf vzniká tak, že se bod řezu pohybuje podél první osy vlastního vektoru a vynáší se hodnota τ^2 (v tomto případě představuje nejlepší výsledek oblast ve středu grafu). Pro názornost převzato z [15].

V poslední fázi své činnosti algoritmus provádí dodatečné filtrování kandidátních bisekcí podle čtyř dalších aspektů v následujícím pořadí, aby zaručil požadované vlastnosti výsledných domén (globularitu):

1. Vysoce flexibilní jednotky s $\tau^2 > 2{,}6\ ps^2$ jsou vždy rozděleny.

2. β-listy tvořící vysoce kooperativní sítě nejsou nikdy rozděleny (žádné residuum, by nemělo mít vodíkovou vazbu s jiným residuem ve stejné doméně a s dalším v jiné doméně).
3. Rozdělení je akceptováno, pokud obě části jsou dostatečně kompaktní (hodnota kriteria kompaktnosti (4) $\gamma > 0{,}8$)
4. Rozdělení, které vytváří dvě malé (menší než 40 residuí) neglobulární jednotky, je akceptováno za podmínky, že rekurzivní aplikace filtrů vytváří dvě domény jedné větší jednotky.

Pokud je jeden z předchozích filtrů splněn, provede se příslušné rozhodnutí a vyhodnocování dále nepokračuje.

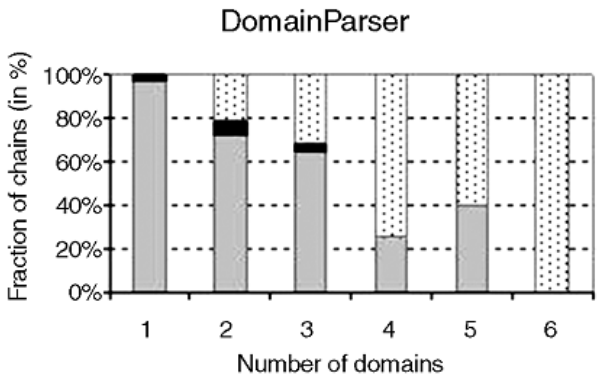
$$\gamma = \frac{1}{N} \sum_{i} \sum_{j<i-3} a_{ij} \tag{4}$$

Vztah (4) představuje kritérium globularity, které je definováno jako síla sekvenčně vzdálených mezi-atomových kontaktů. N představuje počet těžkých atomů v jednotce, a_{ij} sílu kontaktu mezi residui i, j. První až třetí sekvenční soused daného residua i byl z výpočtu vynechán z důvodu zvýšení diskriminace, protože je pravděpodobné, že má být zachován v rozbaleném řetězci.

4.1.2 Domain parser

Domain parser představuje dalšího typického zástupce metod pracujících s informací o vnitřní struktuře proteinu. Hlavní myšlenka algoritmu opět vychází ze základní úvahy popsané v úvodní části kapitoly a spočívá v nalezení takové dekompozice vstupní struktury proteinu, která by obsahovala maximum atomových kontaktů v rámci vzniklých podjednotek a minimum kontaktů mezi jednotkami. Metoda je představena v [22], nezávislé testování na srovnávací sadě je pak provedeno v [11]. Dosažené výsledky pro různé typy topologií dokumentuje obrázek 4.6.

Algoritmus využívá oproti předchozí metodě naprosto odlišný přístup k nalezení nejvýhodnější dekompozice struktury založený na grafových algoritmech (Ford-Fulkerson). V následujícím textu budou postupně nastíněny jeho jednotlivé kroky.

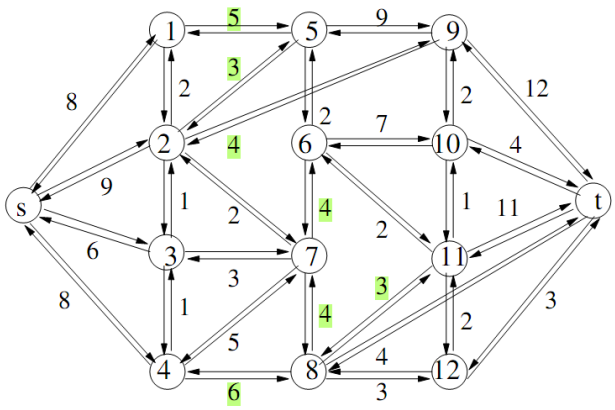


Obrázek 4.6: Hodnocení metody převzaté z [11]. Šedá část představuje úspěšnou predikci, černá část podíl struktur, kde byl predikován větší počet domén a tečkovaná naopak struktury, u kterých byl predikován nižší počet domén.

Vstupem metody je orientovaný graf (graf toku) vzniklý převodem vnitřní trojrozměrné struktury proteinu. Zmíněný graf je tvořen následujícími částmi.

- Uzly představující jednotlivá residua.
- Hranami představujícími residua, která spolu prostorově interagují. K určení této interakce je použito následující kritérium. Jestliže je možné nalézt dvojici atomů (z obou residuí), jejichž vzdálenost je menší nebo rovna 4Å, pak tato residua spolu interagují.
- Speciálními uzly s (source), t (sink), které jsou dále využity algoritmem Ford-Fulkerson.

Každá nalezená spojnice residuí u, v je převedena na dvojici orientovaných hran (u, v) a (v, u). Pro každou takovou hranu (u, v) metoda nejprve stanoví její kladnou kapacitu $c(u, v)$.



Obrázek 4.7: Na obrázku je příklad grafu (struktury proteinu) převzatý z [22]. Zeleně jsou vyznačeny hrany tvořící minimální s-t řez.

Přiřazení kapacity hranám představuje jeden ze základních problémů algoritmu, který má výrazný dopad na kvalitu výsledného rozdělení struktury proteinu mezi jednotlivé domény. K výpočtu kapacity hrany se používá funkce (5), která obsahuje celou řadu parametrů. Optimální hodnota

většiny zmíněných parametrů byla stanovena experimentálně na trénovací sadě dat užitím metody pravoúhlých polí tak, aby respektovala následující body.

- Kontakty v rámci proteinové kostry by měly být přerušovány co nejméně.
- Dekompozice by neměla rozdělovat β-list mezi různé domény.
- β-vlákno by mělo být rozdělováno co nejméně (ideálně pouze v případě all-β proteinů)

$$c(u,v) = k_{u,v} + k_{u,v}^{b}\omega_b + k_{u,v}^{\beta}\omega_\beta + k_{u,v}^{e}\omega_e \tag{5}$$

V rovnici (5) představuje $k_{u,v}$ počet atomových kontaktů mezi residui u a v, $k_{u,v}^{b}$ počet kontaktů mezi atomy polypeptidové kostry (mezi residui u a v). Parametr $k_{u,v}^{\beta}$ nabývá hodnoty 1, pokud u a v vytváří vodíkový můstek mezi atomy kostry napříč β-listem, jinak nabývá hodnoty 0. Podobně nabývá parametr $k_{u,v}^{e}$ hodnoty 1, pokud u a v náleží do stejného β-vlákna. Rovnice dále obsahuje kalibrační konstanty $\omega_b, \omega_\beta, \omega_e$ určené experimentálně.

Vstupní graf toku metoda využívá k rozdělení struktury proteinu na dvě části (stejně jako u předchozí metody dochází k bisekci) tak, aby celková kapacita hran v místech dělení byla minimální. Problém se tedy redukuje na hledání úzkého místa sítě (minimálního s-t řezu) pomocí algoritmu Ford-Fulkerson. Abych mohl naznačit jeho řešení tak, jak je popsáno ve [22], musím nejprve zavést několik pojmů.

Prvním z nich je *s-t řez*, což je množina hran, jejichž odebrání vede k tomu, že z uzlu s přestane existovat cesta do uzlu t. *Minimální s-t* řez je pak řez s nejmenší kapacitou odebraných hran (na obrázku 4.7 jsou jeho hrany vyznačeny zeleně). Výpočet minimálního s-t řezu lze provést nalezením maximálního toku.

Na předchozí specifikaci řezů navazuje definice toku, kdy *s-t tok* je množina hodnot přiřazených hranám orientovaného grafu (reprezentovaná pomocí $f(u,v)$), které splňují následující vlastnosti (pozn.: $c(u,v)$ označuje kapacitu hrany).

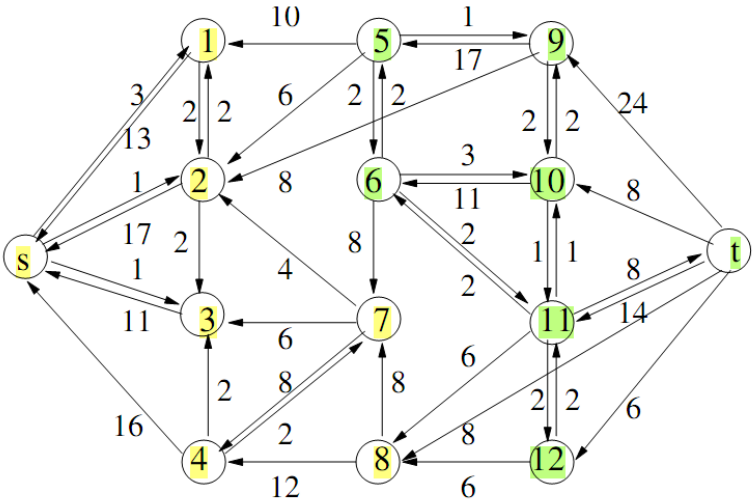
- $f(u,v) \leq c(u,v)$
- $f(u,v) = -f(v,u)$
- $\sum_v f(u,v) = 0$

Cílem zkoumané metody je nalézt maximální tok (který odpovídá minimálnímu řezu) za pomocí algoritmu Ford-Fulkerson. Pro tento algoritmus je zásadní definice tzv. *zbytkové kapacity hrany* (u,v), $c_f(u,v) = c(u,v) - f(u,v)$. Zbytková kapacita hrany nabývá dle předchozího předpisu kladných hodnot, někdy dokonce vyšších než byla původní kapacita (vlivem záporné hodnoty toku).

Důležitým krokem, který algoritmus musí provést před započetím hledání minimálního řezu, je určení uzlů s a t, které ke své činnosti potřebuje algoritmus Ford-Fulkerson. Tyto nejsou voleny přímo z uzlů sítě (což by mohlo vést k následnému triviálnímu rozdělení sítě), ale je využita jiná

strategie, jež vybírá a napojuje dvě skupiny původních uzlů grafu na dodatečně přidané uzly s a t orientovanými hranami s kapacitou $+\infty$ (aby předešla triviálním dělení sítě).

Nyní je možné popsat, jak je v dané metodě realizován zmíněný algoritmus pro hledání maximálního toku. Základní myšlenka spočívá v opakovaném hledání orientované cesty p mezi vrcholy s a t. Nalezená cesta sestává z takových orientovaných hran (u, v), že $c_f(u, v) > 0$. Po nalezení cesty p je u všech jejích hran navýšena hodnota toku f o minimální hodnotu c_f v p (a současně upravena hodnota toku pro (v, u) z důvodu zachování požadovaných vlastností toku). Naznačená procedura se opakuje až do chvíle, než nastane situace, kdy nelze cestu p v grafu nalézt. Zkoumaný algoritmus dále využívá optimalizaci, která snižuje časovou složitost popsané procedury v podobě jednoduchého omezení kladeného na nalezenou cestu p, která musí mít nejnižší počet hran ze všech možných p (nejmenší cesta je nalezena průchodem grafu do hloubky).



Obrázek 4.8: Na obrázku je znázorněna zbytková síť pro nalezený maximální s-t tok, kde jsou vyznačeny kapacity jednotlivých hran. Uzly, které mají spojení s s, jsou označeny žlutě, uzly se spojením s t jsou označeny zeleně. Spojení mezi těmito dvěma skupinami na obrázku není (má kapacitu 0 a jeho hrany tvoří minimální řez). V upravené podobě převzato z [22].

Postup popsaný v předchozí části vytvoří zbytkovou síť pro nalezený maximální tok (zbytková síť obsahuje kladné zbytkové kapacity). Z této sítě je možné získat minimální řez tak, že nejprve vytvoříme dvě skupiny uzlů.

- Do skupiny T (na obrázku 4.8 vyznačena zeleně) zařadíme uzly, ze kterých se lze dostat do uzlu t (sink).
- Do skupiny $\bar{T}$ zařadíme zbylé uzly.

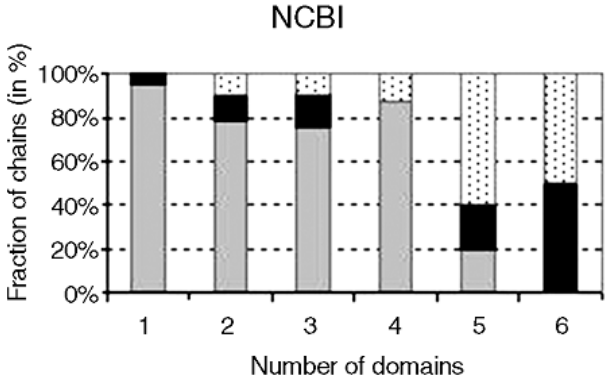
Hrany mezi těmito skupinami mají zbytkovou kapacitu 0 (nejsou zakresleny na obrázku 4.8) a tvoří minimální řez (resp. může existovat více minimálních řezů). Minimální řez původní sítě (na obrázku 4.7) je tvořen těmito hranami, konkrétně {(1,5), (2,5), (2,9), (6,7), (8, 11), (8, 12), (8, t)}. Kapacita

hran zmíněného řezu je 26, což odpovídá maximálnímu nalezenému s-t toku. Odebrání hran řezu vede k rozdělení grafu a také proteinu na dvě části, které přímo odpovídají hledaným doménám. Postup, který je zde popsán, je iterativně opakován (počínaje vytvořením sítě a konče nalezením dělícího řezu) pro obě nalezené domény s cílem určit možnou přítomnost dalších domén. Celý proces končí, pokud obsahují rozdělené části méně než 80 residuí, nebo dojde k jeho zastavení během dodatečné kontroly nalezených struktur.

Jak bylo naznačeno výše, síť může obsahovat více minimálních s-t řezů, algoritmus pak vybírá nejvhodnější z nich v závěrečné fázi své činnosti, kdy provádí dodatečnou korekci výsledků, kterou nyní popíši. Prvním krokem tohoto procesu je ověření vlastností nalezeného rozdělení, kontrolují se především kompaktnost, velikost struktur, počty segmentů, velikost rozhraní mezi doménami a případná rozdělení β-listů mezi nalezené jednotky. Následně může dojít k připojením kratších segmentů a segmentů na koncích proteinového řetězce k přilehlým doménám. Tato fáze končí výběrem nejvhodnějšího rozdělení vstupní struktury.

4.1.3 Metoda NCBI

Poslední zástupce zde popsaných metod pracujících se strukturní informací opět provádí dekompozici proteinu na jednotlivé domény pomocí výše popsaného principu (maximalizace kontaktů v rámci domény a minimalizace kontaktů mezi doménami). Algoritmus, stejně jako jeho předchůdci, postupuje metodou od shora dolů, kdy začíná s celou molekulou a končí jejím rozkladem na výsledné strukturální podjednotky. Výsledky, kterých tento postup dosahuje, byly opět podrobeny srovnávání v [11] a jsou zobrazeny na spodním obrázku. Informace o fungování metody jsem čerpal z [23].



Obrázek 4.9: Hodnocení metody převzaté z [11]. Šedá část představuje úspěšnou predikci. Černá část představuje podíl struktur, kde byl predikován větší počet domén, tečkovaná část představuje naopak struktury, u kterých byl predikován nižší počet domén.

Základem algoritmu je postup známý jako navlékání (threading), který spočívá v nalezení známé struktury (uložené v lokální databázi) ve zkoumaném proteinu. Tento protein je "protahován"

přes jednotlivé vzory databáze a je vyšetřováno jejich vzájemné zarovnání pomocí optimalizace výstupu speciálně navržené skórovací funkce. V následující části textu budou nastíněny jednotlivé hlavní kroky algoritmu tak, jak jsou popsány v [23].

Velmi důležitým aspektem metody a obecně všech postupů pracujících technikou navlékání, je způsob, jakým provádějí konstrukci interní databáze a který následně podmiňuje použitý model zarovnání. Algoritmus vytváří databázi s ohledem na několik základních principů, mezi nedůležitější z nich patří předpoklad existence tzv. jádra proteinové domény. Toto jádro je tvořeno většími prvky sekundární struktury, jejichž konformace je obvykle evolučně konzervována, a spojujícími smyčkami, jejichž délka i prostorové uspořádání je naopak značně proměnlivé. Databáze pak sestává právě z těchto jader doplněných o popis jejich skládání (foldu), který je definován na základě podobnosti mezi strukturami obsahujícími stejné jádro. Prvky sekundární struktury, použité při konstrukci databáze, metoda vyhledá pomocí algoritmu Kundrot-Richards.

Jednotlivé motivy tvořící základ databáze prochází procesem dalšího zpracování, který doplňuje strukturu záznamů o informace vylepšující vlastnosti procesu zarovnání a zároveň redukuje méně kvalitní vzory. Mezi tyto dodatečné kroky patří.

- Doplnění hranice domén, k tomuto účelu je použita metoda podobná s PUU.
- Kontrola, zda doména neobsahuje izolované prvky sekundární struktury.
- Kontrola počtu residuí, které jsou součástí sekundárních struktur.
- Oříznutí residuí z N a C konců sekundárních struktur.
- Vypuštění menších sub-struktur, tj. takových, které obsahují méně než 2 residua ve vlákně (strand), nebo 4 ve spirále.
- Výpočet omezení délky nekonzervovaných částí struktury (reprezentujících smyčky).

Po vytvoření databáze je metoda připravená k predikci domén ve zkoumané struktuře, pomocí postupu známého jako navlékání., během něhož je vstupní sekvence zarovnávána s jednotlivými motivy jader domén s využitím speciálně navrženého modelu. Optimálního zarovnání je dosaženo využitím Gibbsova vzorkovacího algoritmu, dále nejsou penalizovány mezery a není preferována specifická délka vyhledávaných motivů (není tedy primárně vyhledáváno zarovnání s největším motivem).

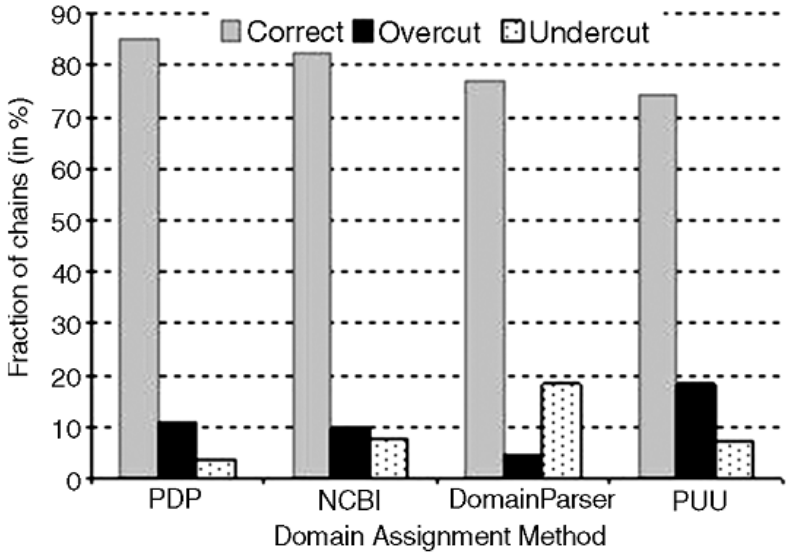
Zmíněná metoda vzorkování provádí iterativní prodlužování nalezených jader přidáváním nových struktur na jejich konce s využití stochastického procesu. Po průchodu databází (resp. zpracování vstupní proteinové struktury) je vybráno 50 nejlepších kandidátních dekompozic, pro něž jsou zaznamenány:

- Energie vypočítaná během navlékání, pomocí funkce definované v [23]
- Skóre korigující celkovou kompozici.

Z kandidátních řešení je následně vybráno takové, které nejlépe uspokojuje požadavky na obě zaznamenané hodnoty.

4.1.4 Souhrnné hodnocení popsaných metod

Na závěr této části textu, věnující se metodám pracujícím s informací o vnitřní struktuře proteinu, uvádím pro zajímavost jimi dosažené výsledky během souhrnného testování publikovaného v [11].



Obrázek 4.10: Porovnání převzaté z [11], představující výsledky testování předchozích metod na srovnávací sadě dat. Šedé části představují procento správně predikovaných struktur, černé procento struktur s menším počtem nalezených domén a tečkované procento struktur s vyšším počtem nalezených domén.

4.2 Metody pracující s informací o sekvenci

V další části kapitoly se zaměřím na skupinu metod pracujících se sekvencí aminokyselin reprezentující protein. Problém predikce proteinových domén založený pouze na znalosti sekvence přináší mnohé dodatečné komplikace. Jednou z hlavních je, že i značně rozdílné řetězce aminokyselin mohou v prostoru nabývat podobné konformace (dle [11] mohou molekuly nabývat obdobné trojrozměrné struktury běžně i při shodě sekvencí nižší než 30%), což téměř znemožňuje vyhledávat prostorové motivy pouze na základě podobnosti jejich sekvencí. Další překážku představuje velká variabilita samotných proteinů, která může vést k existenci domén, jež v různých aspektech porušují obecnou představu o jejich podobě (nemusí být například přítomno hydrofobní jádro, nebo struktura nemusí nabývat globulárního tvaru). Nepřekonatelnou překážku, pro téměř všechny zástupce metod tohoto typu, pak představují tzv. nesouvislé domény (prostorově blízká residua si nemusí být blízká i

v rámci řetězce). V [11] byl kromě zmíněných poukázán další, tentokrát spíš metodický problém, který se stejně jako u předchozí skupiny metod týká chybějícího systematického přístupu, jež by umožňoval porovnávat výsledky jednotlivých algoritmů. Problémy by se daly shrnout do těchto kategorií.

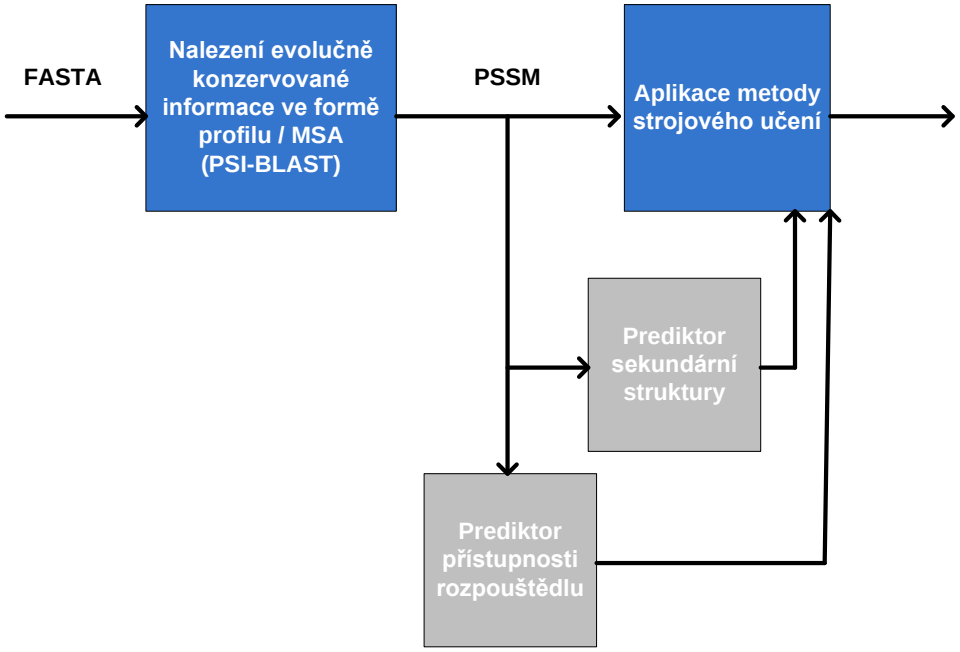
- Standardizovaná testovací sada. Jednotlivé postupy totiž využívají toho, že je volně dostupné dostatečné množství sekvencí a provádí testování na specificky zvolených sadách dat.
- Metodika hodnocení úspěšnosti predikce, kde existuje několik přístupů, mezi nimiž jsou například.
    - Překryv hranice domén, resp. se jako úspěch považuje, pokud se metoda "strefí" do určité tolerance (obvykle do vzdálenosti +/- 10, nebo 20 residuí od skutečné hranice).
    - Překryv domén jako struktur, který sice představuje méně častý způsob hodnocení úspěchu, ale v [11] je považován za nejvíce objektivní.

Oproti strukturním metodám je, ale obvykle využit některý z validních postupů hodnocení úspěšnosti prediktorů – častá je zejména křížová validace.

Přes problémy popsané výše je i této skupině metod věnována poměrně značná pozornost. Přináší totiž jednu nespornou výhodu - ke své činnosti, jak ostatně plyne z názvu, nepotřebuje znát strukturu proteinů. Prostorové uspořádání je totiž zatím určeno pouze u minoritní části známých molekul.

Většina zástupců této kategorie je v současné době různou měrou limitována zmíněnými obtížemi. Obvykle také platí, že se predikce zaměřuje (resp. dosahuje nejlepších výsledků) na proteiny se dvěma doménami. Toto omezení plyne ze základního postupu, který je v různé podobě metodami daného typu využíván – konkrétně jsou v sekvenci vyhledávány oblasti známé jako mezidoménové linkery (předěly, nebo také hraniční oblasti mezi doménami), pro které bylo experimentálně dokázáno (zmínka například v [11]), že se ve většině případů dají dobře odlišit od zbytku sekvence. Jednotlivé algoritmy pak různým způsobem rozvíjejí předchozí postup, například začlenění informace o tom, jak jsou jednotlivá residua přístupná rozpouštědlu (což ve svém principu pomáhá hledat hydrofobní jádra), nebo využitím výstupů poměrně přesné predikce sekundární struktury (pracujících také nad sekvencí). S využitím předchozího konceptu pak nejlepší zástupci této kategorie dosahují až 70% úspěšnosti při predikce domén u proteinů se dvěma doménami. Pro strukturně komplikovanější proteiny tato úspěšnost ovšem dramaticky klesá.

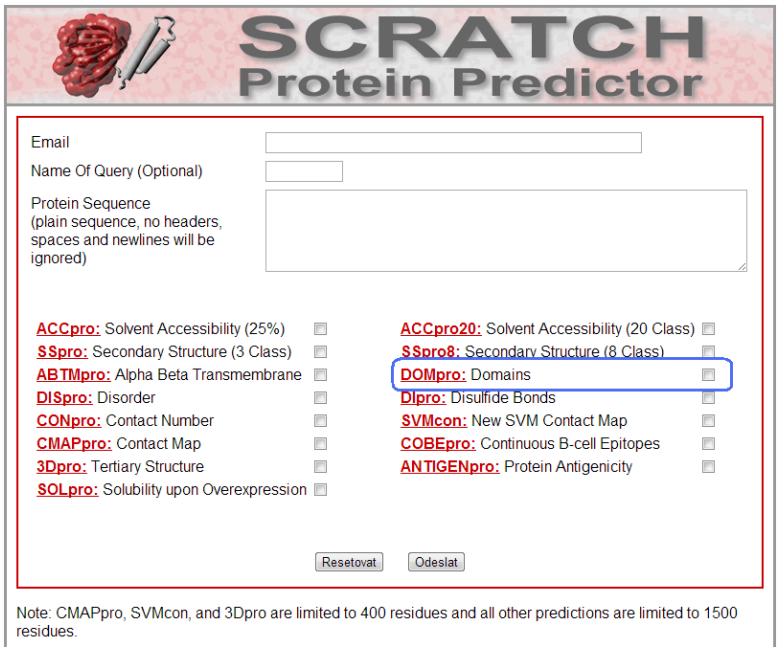
V rámci dalšího textu se zaměřím na zástupce této skupiny, kteří detekují hraniční oblasti domén s využitím metod strojového učení, a to především z toho důvodu, že podobnou metodu jsem se rozhodl implementovat. Typické pro algoritmy tohoto typu je, že pracují podle schématu znázorněnému na spodním obrázku.



Obrázek 4.11: Znázornění modelu činnosti metod predikujících proteinové domény ze sekvence s využitím postupů strojového učení. Modré bloky představují společný základ většiny algoritmů, šedé bloky znázorňují volitelné kroky doplňující informaci o struktuře, popřípadě prezenci hydrofobních residuí.

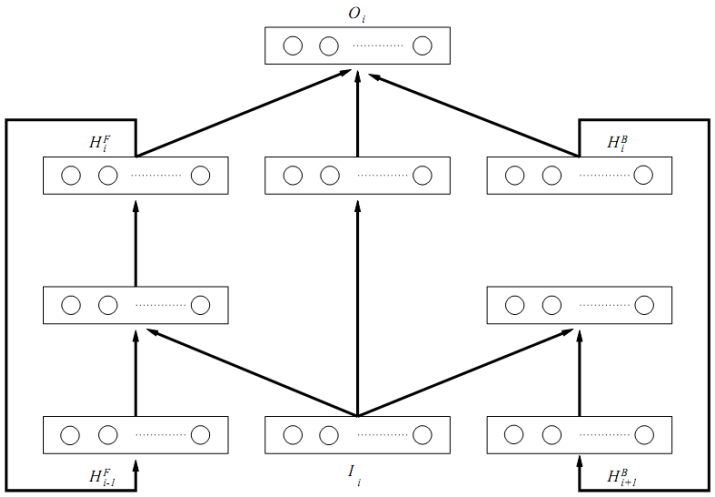
4.2.1 DOMPro

DOMPro představuje prvního zde popsaného zástupce metod pracujících pouze s informací o sekvenci proteinu. Řadí se mezi nejúspěšnější algoritmy tohoto typu. Dle autorů materiálu [19], ze kterého jsem čerpal informace pro tuto část textu, dosahuje dále popsaný postup až 70% úspěšnosti predikce (v testu plně automatické predikce CAFASP-4). Celkově se jedná o mladší metodu pocházející z roku 2005, tvořící součást serveru SCRACTH zaměřeného na oblast predikce sekundárních a terciárních struktur proteinů dostupného na adrese: http://scratch.proteomics.ics.uci.edu/. Uživatelské rozhraní zmíněného nástroje je pro zajímavost zobrazeno na následujícím obrázku.



Obrázek 3.12: Uživatelské rozhraní serveru SCRATCH, jehož součástí je predikce proteinových domén s využitím algoritmu DOMpro (tato volba je označena modře). Výsledek predikce je uživateli přeposlán pomocí e-mailu.

Algoritmus pracuje při predikci na základě postupu, který byl naznačen na předchozí straně. Hlavním principem, který využívá, je hledání hraničních oblastí domén, u nichž předpokládá odlišnou míru evoluční konzervace oproti zbytku sekvence. K detekci těchto útvarů používá, jak bylo řečeno v úvodu této části, metodu strojového učení, v tomto případě založenou na poměrně netradiční jednorozměrné rekurzivní neuronové síti (1D-RNN).

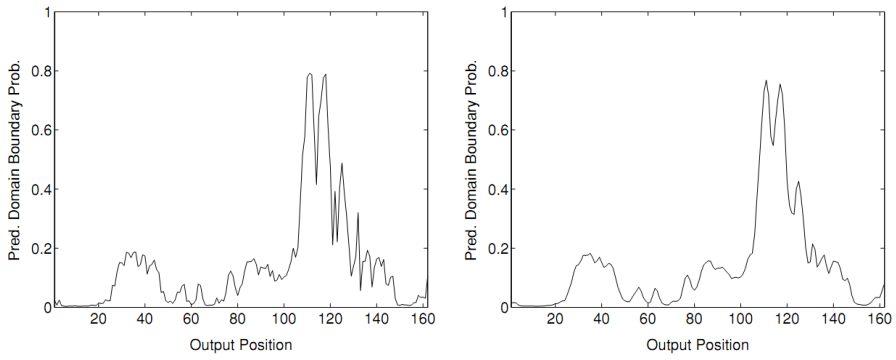


Obrázek 4.13: Příklad architektury jednorozměrné rekurzivní (obousměrné) neuronové sítě převzatý z materiálu [24], ze kterého čerpali autoři právě zkoumaného algoritmu. Síť na obrázku využívá tři typy běžných dopředných neuronových sítí. První z nich zpracovává vstup *I* a produkuje výstup *O*, dvě zbylé sítě zpracovávají dopředné proměnné H_i^F, resp. volitelně zpětné proměnné H_i^B.

První fází, kterou metoda realizuje, je příprava vnitřního modelu, konkrétně trénování vah použité neuronové sítě. K tomuto účelu je vytvořena sada dat obsahující proteiny s několika doménami pocházející z databáze CATH. Sekvence z této databáze obsahují pouze oblasti domén, proto jsou nejprve řetězce rekonstruovány s použitím dat z databáze PDB. Dalším krokem je zjištění vlivu evoluční informace, kterého je dosaženo vytvořením profilů lokálního zarovnání sekvencí k neredundantní databázi (NR) metodou PSI-BLAST. Pro jednotlivé sekvence jsou poté predikovány metodou ab-inito sekundární struktury (nástrojem SSpro) a přístupnosti rozpouštědlu (nástrojem ACCpro). Posledním krokem je samotné trénování neuronové sítě. Problém predikce hranice proteinové domény je zde reprezentován binární klasifikací nad jednorozměrným řetězcem aminokyselin. Residua náležící do oblasti hranice jsou vybírána na základě vzdálenosti od skutečné hranice, přesněji residua ve vzdálenosti +/- 20 pozic od skutečné hranice jsou považována za hraniční a ostatní za nehraniční. Pokud toto shrneme, vstupem neuronové sítě během procesu učení jsou vektory obsahující následující položky.

- 21 reálných hodnot reprezentujících pravděpodobnosti záměn aminokyselin pro danou pozici řetězce, vzatých z vytvořeného profilu.
- Tři binární hodnoty reprezentující predikovanou strukturní třídu aktuálního residua (spirála, vlákno, nebo svinutí).
- Dvě binární hodnoty, obsahující výsledek predikce určující přístupnost residua rozpouštědlu.
- Binární hodnota reprezentující požadovaný výstup klasifikátoru (1 – hraniční oblast, 0 – nehraniční oblast).

Během predikce nových struktur pracuje algoritmus obdobným způsobem, provádí generování profilu pomocí lokálního zarovnání, následně pro jednotlivá residua predikuje třídu sekundární struktury a přístupnost rozpouštědlu. Poté jsou tato data poslána na vstup neuronové sítě a její odezva je pro jednotlivá residua zaznamenána.

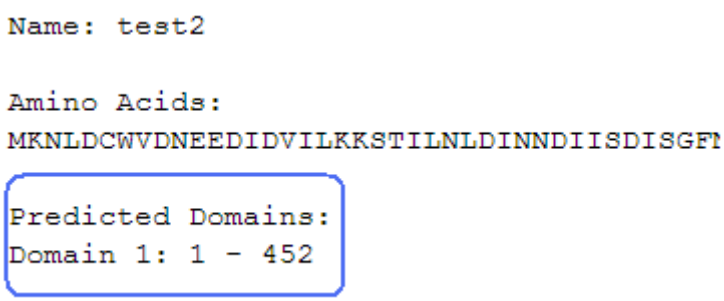


Obrázek 4.14: Vliv vyhlazení (pravá část) původního výstupu sítě (vlevo) oknem o velikosti 3 residua.

Nyní přichází na řadu další zpracování výstupu sítě. Prvním krokem je vyhlazení dat, které je realizováno průměrování v rámci okna o velikosti tři residua. Dosažený efekt je zdokumentován na obrázku 4.14. Výstup po vyhlazení stále ještě není úplně vhodný pro přímé určení hranice, proto je pozice hranice a následné rozdělení sekvence odvozeno tímto procesem.

1. Ve výstupních datech jsou detekována kandidátní místa, vyhledáním vzoru $(B + N\{0, m\} + B+)$. B zde představuje residuum označené jako hraniční, N residuum označené jako nehraniční a m maximální délku rozdělení dvou hraničních residuí.
2. Jsou vyřazeny hraniční regiony kratší než tři residua.
3. Sekvence je rozdělena uprostřed nalezených hraničních oblastí a jednotlivé segmenty jsou přiřazeny k separátním doménám.

Výsledek predikce je uživateli prezentován v podobě, která je znázorněna na následujícím obrázku představující výstup reálné implementace metody DOMpro, která tvoří součást zmíněného serveru SCRATCH.



Obrázek 4.15: Formát výstupu metody DOMpro. Uživateli je výsledek predikce doručen pomocí e-mailu, predikované struktury jsou znázorněny modře. Celková délka predikce v testovaném případě (sekvence o délce 453 residuí obsahující dvě domény s hranicí na pozici 247) byla 20 minut.

4.2.2 DoBo

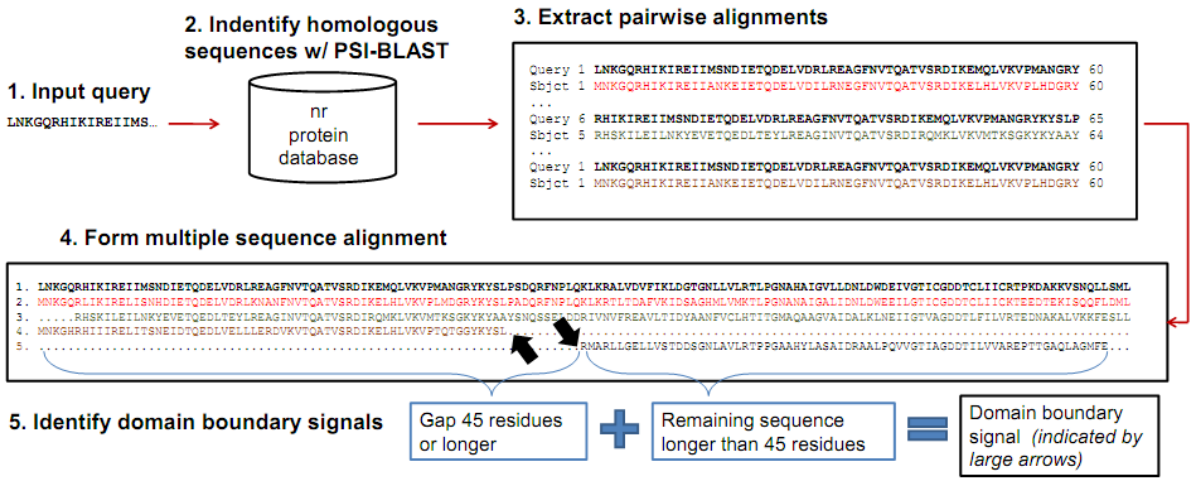
DoBo je dalším postupem využívajícím k predikci proteinových domén pouze informaci o primární struktuře proteinu. Zároveň představuje úplně nejmladší zde prezentovanou metodu, která byla publikována v [14] v roce 2011. V mnoha aspektech vychází tento algoritmus z předchozího zástupce (DOMpro), hlavním rozdílem mezi nimi je především postup strojového učení, který je v tomto případě reprezentován metodou SVM. Opět je integrována evoluční informace (v rámci [14] nazývaná evoluční signál) získaná s pomocí nástroje PSI-BLAST, doplněná o předpovězenou sekundární strukturu a přístupnost rozpouštědlu. Predikce domén pak spočívá, stejně jako u předchozího algoritmu, ve vyhledávání specifických oblastí sekvence, které jsou označovány jako mezidoménové hranice (linkery).

V první fázi své činnosti musí algoritmus natrénovat použitou metodu strojového učení, aby rozpoznávala hraniční oblasti domén. K tomuto účelu používá sadu dat připravenou pro trénování a evaluaci předchozího zástupce. Ve zmíněné sadě však ponechává pouze proteiny, jejichž struktura je učena shodně v databázích SCOP i CATH. Dále vybírá proteiny, které mají délku větší než 90 residuí, aby se mohly během multisekvečního zarovnání projevit evoluční signály hraničních oblastí.

Hranice domény, resp. evoluční signál, je v rámci algoritmu detekována pomocí metody PSI-BLAST. Sekvence je nejdříve vyhledána v neredundantní (NR) databázi, následně je vytvořeno multisekvenční zarovnání (MSA) podle vstupní sekvence. Poté jsou v nalezených zarovnáních (tj. úsecích proteinových řetězců) detekovány hraniční oblasti podle následujících pravidel.

- Hraniční oblast je definována jako souvislá mezera začínající na N, nebo C konci daného fragmentu o délce minimálně 45 residuí.
- Po odstranění zmíněné mezery musí zbytek sekvence obsahovat alespoň 45 residuí.
- Hranici pak představuje první residuum, které netvoří mezeru.

Jednotlivé takto nalezené hraniční oblasti jsou shromažďovány, dokud nejsou zpracovány všechny sekvence v MSA, nebo nebyla hranice nalezena na 35 různých místech vstupní sekvence. Poté jsou posbírané pozice klasifikovány do jedné ze tří tříd (falešná hranice, blízká hranice, daleko od hranice).



Obrázek 4.16: Znázornění popsaného procesu zpracování evolučního signálu - od vytvoření multisekvenčního zarovnání po detekci konkrétní hranice. Převzato z [14].

Hranice proteinové domény je opět definována jako oblast +/- 20 residuí od reálné hranice (tj. od hranice definované v tréninkové sadě dat). K její predikci je použita trojice binárních klasifikátorů SVM (v implementaci SVMlight).

- První z nich klasifikuje vstup do třídy falešná hranice. Jako falešná hranice jsou označeny vzory ze sekvencí jednodoménových proteinů.
- Druhý rozděluje vstup podle toho, jestli se nachází v oblasti blízké doméně (+/- 20 residuí) – rozlišuje třídu blízké hranice.
- Třetí naopak rozpoznává, že vstup náleží do oblasti vzdálené doméně (více než +/- 20 residuí) – rozlišuje třídu vzdálené hranice.

Zmíněné klasifikátory jsou aplikovány ve dvou krocích, nejprve provede klasifikaci první z nich. Pokud zařadí vstupní vektor do třídy pravé hranice, jsou aplikovány paralelně zbylé dva a podle toho, jak jejich klasifikace dopadne, je označeno aktuální residuum za hraniční, nebo nehraniční.

Vstupní vektor všech tří klasifikátorů má 1071 složek, které reprezentují hodnoty rozepsané v následujícím seznamu.

- 21 reálných hodnot pro každou pozici v okně o délce 41 residuí centrovaných kolem aktuálně zpracovávané pozice, představující frekvence záměn aminokyselin (resp. mezer) vypočítaných jako profil metody PSI-BLAST.
- 5 binárních hodnot určujících předpovězenou sekundární strukturu a přístupnost rozpouštědlu pro každé z reziduí v aktuálním okně. K predikcím těchto hodnot je využit nástroj SSpro.
- 3 hodnoty, které se vztahují ke zpracovávané pozici (resp. signálu).
    - Pozice zpracovávaného residua vzhledem k N konci (pozice dělena 100).
    - Pozice zpracovávaného residua vzhledem k C konci ((délka proteinu – pozice) / 100).
    - Počet přítomných hraničních oblastí v MSA v sousedství o velikosti pěti residuí.
- Délka proteinu dělená 100.

Po natrénování všech tří klasifikátorů může metoda provádět predikce nad neznámými sekvencemi. Tento proces probíhá podobně jako učení. Pro zkoumanou sekvenci je nejprve vytvořeno MSA. Následně jsou vyhledány možné pozice hranic, a pro ně vytvořeny výše popsané vektory. Nad vektory pak proběhnou dvě fáze klasifikace, které určí, jestli daná pozice představuje doménu. Právě popsaná metoda dosahuje podle autorů [14] 60 % úspěšnosti predikce během 10 cyklů křížové validace na sadě dat shodné té, jaká byla použita pro evaluaci algoritmu DOMpro.

4.3 Závěrečné shrnutí

Na závěr této části textu bych chtěl zdůraznit, že existuje celá řada dalších algoritmů. Mnohé z nich zajímavým způsobem kombinují přístupy popsané výše, v [11] jsou definovány jako metametody. Kromě nich byly vyvinuty další postupy, které se kompletně vymykají obecným principům obou kategorií metod popsaných v této kapitole. Mezi ně lze zařadit například metodu DGS (Domain Guess by Size), jež predikuje proteinové domény pouze na základě velikosti proteinu, jeho segmentů a samotných domén s využitím pravděpodobnostní funkce. Přes poměrně nezvyklý přístup dosahuje DGS poměrně vysoké úspěšnosti predikce (dle [11] přibližne 60%) a zvládá nalézat i nespojité domény.

V následující kapitole bych chtěl navázat na předchozí text rozborem implementované metody, která využívá primární struktury proteinů k predikci proteinových domén založené na detekci hraničních oblastí.

5 Implementovaná metoda

V následující části textu se pokusím využít předchozích informací k výběru vhodného postupu predikce proteinových domén, k jeho důkladnému rozboru a následné implementaci. Po zralé úvaze jsem se rozhodl vyjít z metod pracujících s primární strukturou proteinu, tedy s jeho sekvencí. Důvodů k tomuto rozhodnutí jsem měl několik, prvním z nich byla bezesporná výhoda podobných metod, které ke své činnosti potřebují pouze sekvence proteinů. Dnešní databáze navíc obsahují poměrně značný objem nasekvenovaných molekul, proto není obtížné získat dostatečný objem dat jak pro trénování, tak i testování zvoleného postupu. Další pozitivum spatřuji v komplexnosti řešeného problému, které umožňuje využít poměrně široký okruh různých přístupů a skýtá možnosti dílčích optimalizací.

Při hledání vhodného výchozího algoritmu pracujícího se sekvencemi jsem se rozhodl využít úspěchů metod používajících přístupy strojového učení a vyzkoušet si podobný postup realizovat. V rámci implementace zvolené metody jsem se zaměřil především na proteiny se dvěma doménami, protože při jejich analýze dosahují metody tohoto typu nejlepších výsledků.

Realizovaná implementace, která je dále popsána, vychází z algoritmu PPRODO popsaného v [18] a dostupného z adresy http://gene.kias.re.kr/~jlee/pprodo/. Zmíněný algoritmus představuje svým způsobem etalon metod zvoleného typu. Jako vůbec první přichází se základním konceptem popsaným v sekci 4.2 předchozí kapitoly. V rámci této práce jsem ze zmíněného algoritmu převzal:

- navržený základní postup predikce využívající evoluční konzervovanost hranic domén,
- metodu strojového učení – neuronovou síť, včetně její optimální architektury,
- sadu dat (proteinů se dvěma doménami) pro testování a evaluaci.

Na druhou stranu jsem uplatnil některé dílčí optimalizace a rozšíření zahrnující jak úpravy samotného algoritmu, tak vytvoření vizuálních nástrojů umožňujících uživateli zprostředkovat výsledky predikce. Mezi zmíněné bych zařadil:

- proces čištění dat před vstupem do klasifikátoru,
- vylepšení procesu učení neuronové sítě, kde jsem experimentoval s několika postupy, včetně použití jednoduchého genetického algoritmu pro optimalizaci vah sítě,
- většinu použitých postupů jsem se pokusil optimalizovat pro běh na současných multi-core výpočetních platformách paralelizací výpočetně nejnáročnějších segmentů zdrojového kódu,
- vytvořil jsem uživatelské rozhraní, umožňující uživateli v přehledné formě zobrazovat výsledek predikce,
- vytvořil jsem uživatelské rozhraní umožňující uživateli prezentovat v různé formě výsledky testování naučeného klasifikátoru.

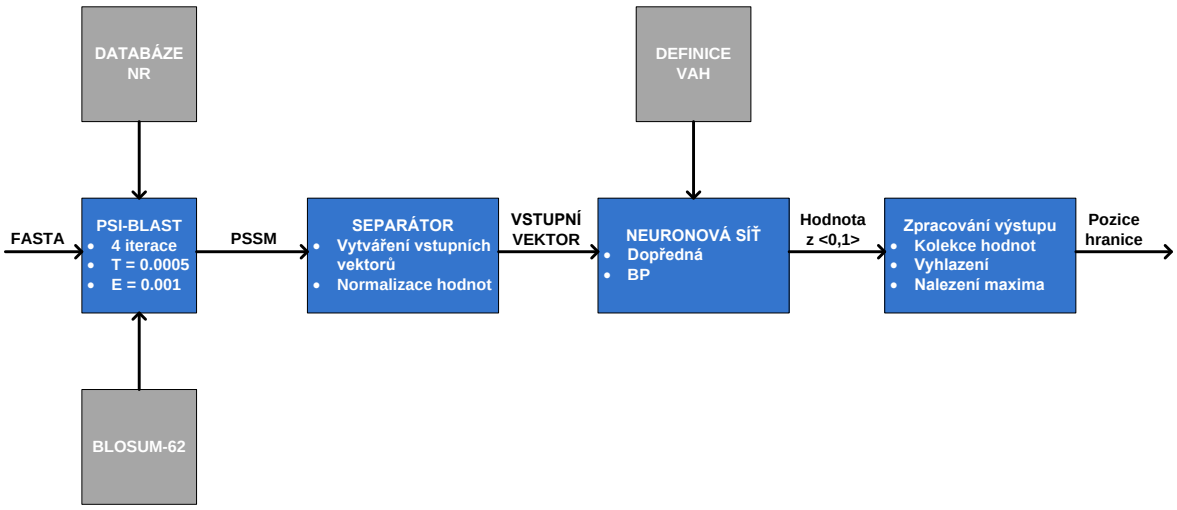
5.1 Návrh algoritmu

Jak bylo řečeno v úvodní části kapitoly, hlavní myšlenky algoritmu vycházejí ze [14]. V následující části textu se zaměřím na jednotlivé aspekty zvoleného postupu.

Implementovaný algoritmus vychází z obecného postupu popsaného v sekci 4.2. Základním principiem využitým při realizaci metody je detekce evolučně konzervovaných úseků v rámci primární struktury proteinu. Tyto úseky vznikají v průběhu let procesem nazývaným přeskupování exonů [25], během něhož dochází k vytváření nových genů spojováním exonů z různých genů ektopickou cestou, nebo duplikování stejného exonu za vzniku nové kombinace exon-intron. Vliv, který tento proces má na hranici proteinů, je dle [14] následující.

1. Residua v oblasti poblíž hranici procházejí často procesem mutace.
2. Residua, která se nacházejí v těsné blízkosti hranice, jsou naopak evolučně silně konzervována, a to i přes častý přesun domén v rámci řetězce.

Úseky, které jsou zmíněny ve druhé odrážce, je pak možné detekovat jako specifické vzory v rámci profilu (PSSM) vytvořeného metodou PSI-BLAST.

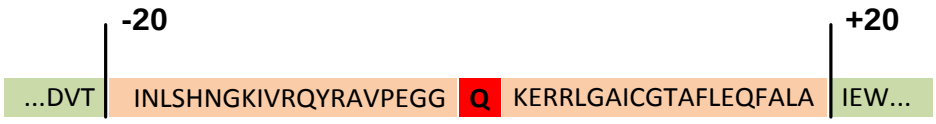


Obrázek 5.1: Neformální znázornění jednotlivých fází zvolené metody. Proces začíná vstupem primární struktury proteinu v podobě sekvence aminokyselin uložené v FASTA souboru. Následuje vytvoření matice PSSM, její dělení na vstupní vektory (pro jednotlivé pozice vstupního řetězce) a klasifikace těchto vektorů pomocí neuronové sítě. Výstup neuronové sítě je pro jednotlivé pozice vstupního řetězce zaznamenáván, vzniklý vektor hodnot je vyhlazen a je v něm nalezeno globální maximu, které reprezentuje nalezenou hranici.

Na předchozím obrázku jsou znázorněny jednotlivé kroky použité metody. Při jejich podrobnějším popisu se zaměřím postupně na dvě hlavní fáze algoritmu, a to učení a samotnou predikci. Nejprve však začnu popisem procesu zpracování vstupních dat, tedy převodu FASTA

sekvence na posloupnost vstupních vektorů neuronové sítě. Tento proces začíná nalezením homologií ve vstupním řetězci s využitím nástroje PSI-BLAST, této metodě se bude důkladněji věnovat příslušná sekce následujícího textu. Výsledkem zpracování sekvence je tzv. profil lokálního zarovnání reprezentovaný maticí PSSM, která pro každé residuum vstupního řetězce určuje frekvence záměn aminokyselin. Vytvořená matice je rozdělena na okna obklopující aktuálně zpracovávané residuum (složená z +/- 10 řádku PSSM), řádky vzniklých oken jsou serializovány do jednoho vektoru o délce 400 hodnot. Složky tohoto vektoru jsou nejdříve normalizovány pomocí sigmoidální funkce, další úpravy dat jsou popsány v sekci 5.3 a upravené hodnoty jsou následně poslány na vstup klasifikátoru.

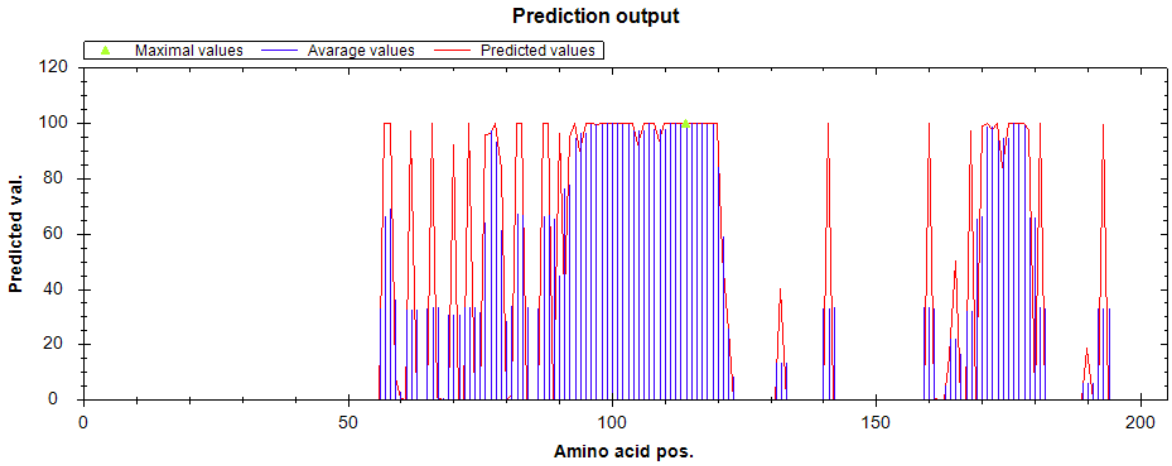
Před samotným trénováním klasifikátorů je nutné nejprve určit, do jaké třídy testovaný vstup reálně náleží. Pro predikci hranic proteinových domén je navíc typické, že pouze malý zlomek z celkového počtu residuí náleží do oblasti hranice (teoreticky 1 residuum / 1 hranice). Na první pohled tak dochází ke značnému nevyvážení trénovacích dat, abych tomuto problému předcházel, tak jsem v souladu s [14] jako hraniční oblast považoval residua v okolí +/- 20 pozic od reálné hranice, zbytek sekvence je považován pochopitelně za oblast nehraniční.



Obrázek 5.2: Na obrázku je znázorněna světle červenou barvou celá hraniční oblast, červenou barvou reálné hraniční residuum a zeleně zbytek řetězce (nehraniční oblast) - pro reálnou sekvenci s FASTA identifikátorem 1b63A.

Data připravená popsaným způsobem jsou dále analyzována klasifikátorem, který je v tomto případě reprezentován dopřednou neuronovou sítí. Síť obsahuje jedinou skrytou vrstvu a jeden výstupní neuron (tj. funguje jako binární klasifikátor). Optimalizace vah sítě probíhá pomocí metody Back-Propagation doplněné o několik vylepšení, která blíže popisuje sekce 5.4. Vstupem sítě je vektor se 400 složkami, reprezentující okolí aktuálně zpracovávaného residua a během procesu učení také požadovaný výstup klasifikace dle předchozí definice (1 – hranice, 0 – oblast mimo hranici).

Po natrénování vah sítě může metoda začít predikovat hranice domén v předkládaných sekvencích proteinů. Postup použitý při predikci odpovídá právě popsanému, výstup sítě pro jednotlivá residua vstupního řetězce je ale shromažďován do výstupního vektoru. Aby se alespoň částečně eliminoval vliv lokálních extrémů neuronové sítě, jsou výstupní data nejprve vyhlazena. V tomto případě pomocí průměrování hodnot v okolí +/- 2 pozice. Ve vyhlazeném výstupním vektoru je následně určena hranice domén jako pozice maximální hodnoty.



Obrázek 5.3: Graf zobrazující reálný výstup prediktoru pro zpracovanou sekvenci. Původní výstup sítě je zobrazen červeně, vyhlazený výstup modrými liniemi (sloupce). Vertikální osa znázorňuje hodnotu výstupu neuronové sítě, horizontální potom jednotlivé pozice reziduí. Trojúhelníkem je vyznačena detekovaná hranice (maximum ve vyhlazeném výstupu).

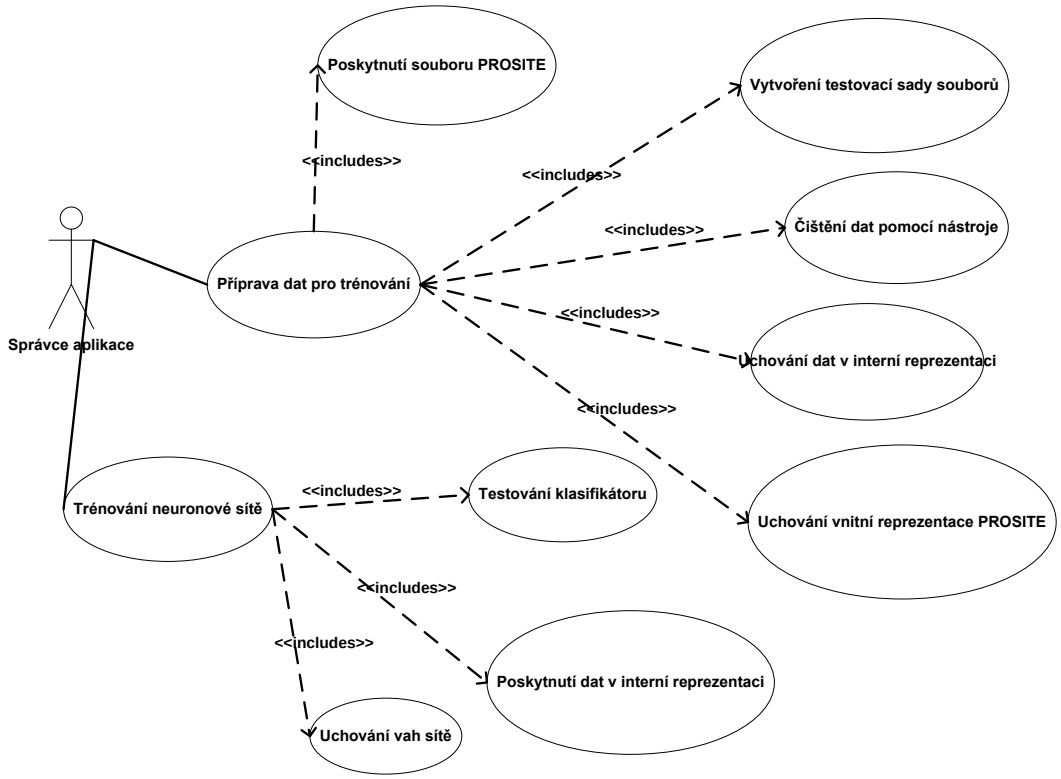
Návrh aplikace z pohledu jejího reálného užití

Během návrhu aplikace jsem vycházel z několika předpokladů o jejím možném skutečném využití. Uživateli toho typu produktu musí být podle mého názoru umožněno provádět:

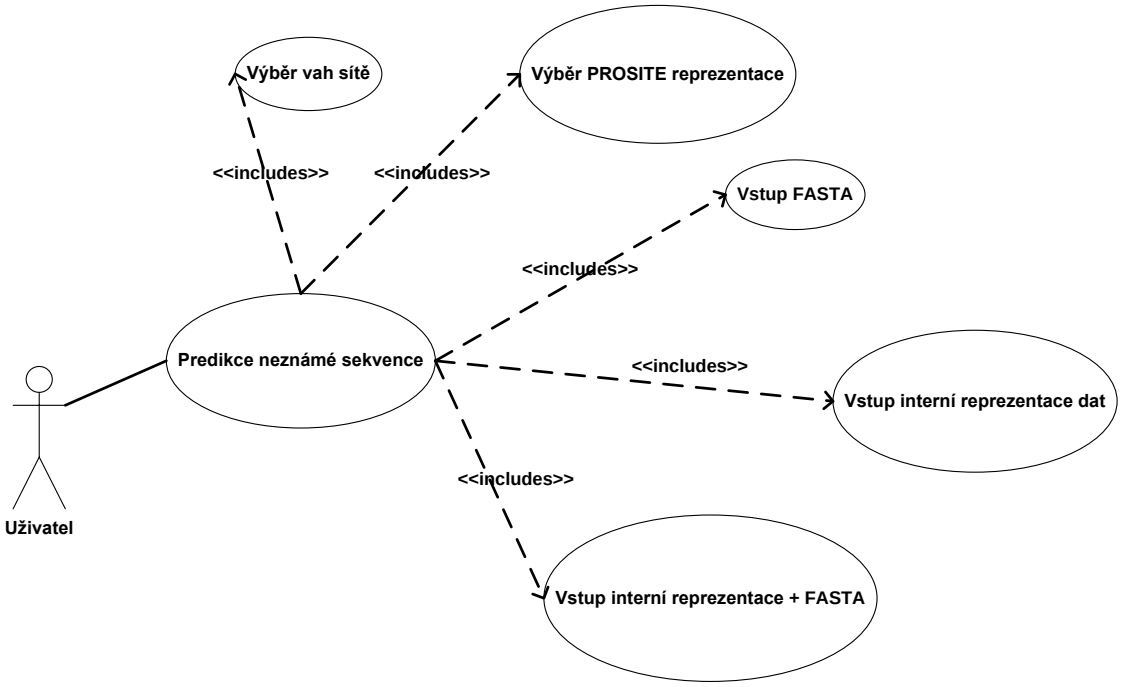
- přípravu dat pro trénování klasifikátoru,
- trénink klasifikátoru a vyhodnocení jeho vlastností jednoduchým způsobem,
- predikce FASTA souborů a vizualizace jejich výsledků.

Jednotlivé výše vyjmenované položky jsem zpracoval do podoby konkrétních nástrojů, které budou popsány v příslušných částech následujícího textu.

K implementaci jsem zvolil prostředí Microsoft .net ve verzi 4.0 (běžně distribuované v operačních systémech Windows Vista a Windows 7), běh aplikací je tímto způsobem zajištěn na běžných pracovních stanicích, jejichž soudobé vlastnosti (především vícejádrové procesory) jsou aplikačně plně podporovány. Výkon, který zvolené prostředí poskytuje, plně postačuje k řádné funkci vytvořených nástrojů. Pokud by však tento model nevyhovoval, je aplikace po návrhové stránce z větší části (vyjma uživatelských rozhraní) připravena pro nasazení v serverových, resp. cloudových řešeních společnosti Microsoft.



Obrázek 5.4: Use-case diagram předpokládaného užití aplikace zkušeným uživatelem k přípravě dat pro trénování klasifikátoru, samotné trénování neuronové a vyhodnocení nalezených vah.



Obrázek 5.5: Use-case diagram přepokládaného užití aplikace běžným uživatelem k predikci neznámé sekvence. Uživateli je umožněno zvolit použité nastavení vah sítě a výběrem vstupu určit použitý režim predikce. Výběr PROSITE reprezentace umožňuje rozšířit možnosti výstupu aplikace a bude dále vysvětlen.

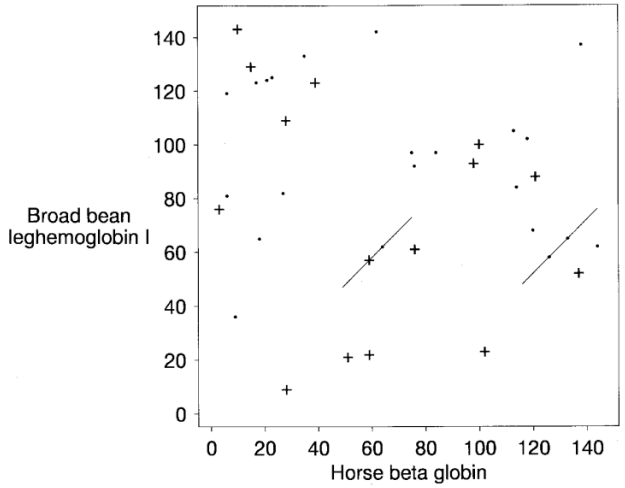
5.2 Metoda PSI-BLAST

V další části textu se zaměřím na poměrně důležitou komponentu implementovaného algoritmu, na metodu PSI-BLAST. Nejprve bych chtěl nastínit principy, které využívá při vyhledávání homologií vstupní sekvence v interní databázi. Následně popíši, jak konkrétně tuto metodu (v implementaci NCBI PSI-BLAST) v rámci realizovaná aplikace využívám.

Algoritmus PSI-BLAST vychází z metod typu BLAST (Basic Local Aligment Search Tool), které vylepšuje z pohledu rychlosti zpracování dotazů, a také senzitivity k méně podobným sekvencím. Jeho primární funkcí je vyhledávat pro danou vstupní posloupnost aminokyselin podobné řetězce v pracovní databázi na základě heuristické metody lokálního zarovnání. Výstupem algoritmu pak může být samotné multisekvenční zarovnání MSA, nebo tzv. PSSM (Position Specific Scooring Matrix), která ve své podstatě představuje frekvence záměn jednotlivých residuí vstupní sekvence v nalezeném MSA. Informace k této části textu jsem čerpal především z [26].

Činnost algoritmu probíhá v několika cyklech, nejprve metoda BLAST v databázi vyhledává slova (nejčastěji o délce tří aminokyselin), jejichž skóre při lokálním zarovnání k vstupní sekvenci překračuje prahovou hodnotu *T*. Při výpočtu zmíněného skóre je použita konkrétní skórovací matice (např. BLOSUM). V další kroku se snaží metoda vyhledat ta zarovnání, která mají dostatečně velké ohodnocení k tomu, aby mohla být reportována na výstupu.

K tomuto účelu je prováděno postupné rozšiřování nalezených fragmentů v obou směrech, během něhož se kontroluje skóre zarovnání. Dle [26] představuje tato fáze algoritmu největší výpočetní zátěž (až 90%). Metoda PSI-BLAST zde provádí dodatečnou optimalizaci, kterou se snaží snížit celkový počet prodlužovaných fragmentů. Zmíněné vylepšení je založeno na výběru pouze těch fragmentů, pro které lze na stejné diagonále (význam diagonály je znázorněn na spodním obrázku) ve vzdálenosti maximálně *A* (bez překryvu) nalézt nějaký další fragment.



Obrázek 5.6: V grafu jsou vyznačeny úseky sekvencí, které při lokálním zarovnání dvojice proteinů vykazují vyšší hodnotu skóre. Diagonály v obrázku znázorňují výše popsanou podmínku rozšiřování.

Po skončení předchozích kroků (a vylepšeného spojování segmentů) je z výstupu algoritmu BLAST vytvořena PSSM (konstrukce matice bude popsána později), odpovídající svoji velikostí délce původní sekvence. Vypočítaná matice je v následujících iteracích běhu metody PSI-BLAST opět použita k vyhledávání podobných sekvencí v databázi. Vzniklý iterativní proces umožňuje dohledávat i poměrně slabě sekvenčně spřízněné proteiny.

Algoritmus dalšího vyhledávání příbuzných sekvencí v databázi opět využívá metodu lokálního zarovnání popsanou výše, jediným rozdílem je vyhodnocování skóre zarovnání, které tentokrát vychází přímo z postupně vytvářené PSSM (není využita skórovací matice).

Jak již bylo řečeno, po každé iteraci běhu metody je vytvářena matice PSSM. Proces, kterým vzniká, je zahájen konstrukcí multisekvenčního zarovnání (MSA), které obsahuje sekvence databáze, pro něž je E hodnota (její výpočet je detailněji popsán v [26]) zarovnání nižší než zvolený práh (implicitně menší než 0,01). Jako vzor při konstrukci MSA je použita vyhledávaná sekvence. Sloupce vzniklého zarovnání jsou použity při konstrukci matice PSSM, která však nezávisí pouze na residuích v nich obsažených, ale i na těch v jejich okolí. Proto je z MSA nejprve zkonstruováno tzv. redukované multisekvenční zarovnání M_C pro každý sloupec C. M_C je sestaveno z množiny sekvencí R, které obsahují nějaké reziduum ve sloupci C a jsou v něm obsaženy pouze ty sloupce (z MSA), ve kterých jsou zastoupeny všechny řetězce z R.

Dále jsou sekvence z MSA váhovány, aby se kompenzoval vliv jejich různých délek. Nakonec dochází k samotnému výpočtu finálních frekvencí záměn residuí v dotazovaném řetězci (vytvářejících PSSM) za pomoci získaného MSA, M_C a vypočtených vah (úplný postup výpočtu je popsán v [26]).

Využití metody PSI-BLAST v rámci implementovaného algoritmu

Jak bylo zmíněno v úvodu kapitoly, výše představenou metodu používám k vytváření profilů (PSSM) ze vstupních sekvencí analyzovaných proteinů v rámci implementovaného postupu. V získaném profilu pak vyhledávám evolučně konzervované úseky pomocí metody strojového učení.

Konkrétně využívám již implementovaný stejnojmenný nástroj, který je součástí balíku BLAST+ volně dostupného z: ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/. Přesné nastavení parametrů metody je popsáno níže.

- Konstrukce PSSM je založena na neredundantní databázi (NR), se kterou dosahuje dle[26] i [18] metoda PSI-BLAST nejlepších výsledků. Databáze je opět volně dostupná z: ftp://ftp.ncbi.nlm.nih.gov/blast/db/FASTA/.
- Skórovací maticí předpokládanou pro první fázi metody jsem v souladu s [18] zvolil BLOSUM62.
- Hodnotu prahu T pro párové zarovnání jsem nastavil na 0,005.
- Hodnotu E jsem nastavil na 0,001.
- Počet iterací metody jsem nastavil na čtyři.

Na následujícím obrázku je znázorněn převod vstupní sekvence proteinu na odpovídající profil PSSM zpracovaný použitým nástrojem. Počet řádků finální matice (už z popisu algoritmu) odpovídá délce vstupní sekvence. Další zpracování znázorněného výstupu je popsáno ve zbylých částech kapitoly.

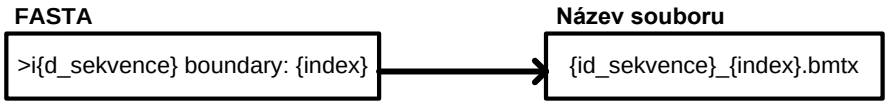
```
PŮVODNÍ FASTA SEKVENCE PROTEINU S IDENTIFIKÁTOREM  1a04A
  E P A T I L L I D D H P M L R ...

NORMALIZOVANÁ PSSM (V ROZSAHU <-10, 10>)
         A  R  N  D  C  Q  E  G  H  I  L  K  M  F  P  S  T  W  Y  V
  1 E   -3 -1 -3 -2 -6  3  5 -4 -3 -5 -5  5 -4 -6 -4 -3 -3 -5 -4 -5
  2 P   -5 -4 -1 -3 -3 -3 -4 -6 -5 -3 -3  3  9 -5  4 -3 -4 -6 -6 -4
  3 A    1 -4 -4 -5 -4 -3 -4 -4 -1  5  0 -1  3 -2 -2  0  3 -5 -2  0
  4 T   -5  5  1 -4 -6 -2 -3 -5  2 -6 -5  5 -5 -7 -5  0  2 -7 -5 -6
  5 I   -4 -7 -8 -8 -3 -7 -7 -8 -8  7  0 -7 -2 -5 -7 -6 -4 -7 -6  5
  6 L   -2 -6 -7 -8 -1 -6 -7 -5 -5  0  6 -6  2  2 -7 -6 -4 -5 -2  1
  7 L   -4 -7 -7 -7 -3 -6 -7 -8 -7  6  3 -7 -1 -3 -7 -5 -3 -6 -5  4
  8 I    2 -7 -7 -7 -1 -6 -7 -6 -7  3 -3 -6 -4 -5 -7 -5 -5 -7 -6  7
  9 D   -7 -6 -4  7 -9 -4  5 -7 -6 -8 -9 -4 -8 -9 -7 -5 -6 -9 -8 -8
 10 D   -5 -7 -3  9 -9 -6 -4 -7 -3 -9 -9 -6 -9 -9 -5 -4 -7-10 -9 -9
 11 H   -6 -5 -5  2 -8  1  5 -7  9 -8 -8 -5 -7 -7 -7 -5 -5 -8 -5 -8
 12 P    2 -2 -3  0 -5  1  2 -3 -1 -4 -3  1 -2 -5  5 -1  0 -6 -5 -2
 13 M    1 -2 -1  3 -4 -4 -3 -2 -4  2  2 -4  1 -3 -1  1 -2 -5 -4  1
 14 L   -4 -6 -7 -7 -5 -2 -6 -7 -6  6  2 -6  1  3 -7 -6 -3 -6 -2  4
 15 R   -1  7 -3 -6  0 -2 -5 -5 -2  0  0 -3 -5 -6 -7  0 -4 -7 -5 -1
 16 T    0 -1 -1  2 -3  1  5 -3 -1 -4 -3  0  1 -5 -4  0  0 -6 -3 -3
 17 G    2 -5 -3 -5 -5 -5 -5  5 -5  0  2 -5  3 -2 -2 -2  0 -6 -5 -2
 ...
```

Obrázek 5.7: Převod vstupní sekvence proteinu na odpovídající normalizovanou matici PSSM metodou PSI-BLAST nastavenou podle výše popsaných zásad.

5.3 Příprava dat pro klasifikátor

Předchozí část kapitoly se zabývala způsobem získání PSSM profilu, tato část textu na ni naváže a předvede, jak je vytvořený profil v aplikaci následně zpracováván. Dále popsaný proces probíhá až na několik výjimek stejně jak ve fázi trénování klasifikátoru, tak i při následné predikci.

Před procesem učení neuronové sítě je trénovací sada FASTA sekvencí se známou hranicí (která bude důkladněji popsána v kapitole zaměřené na testování) postupně předávána metodě PSI-BLAST, která z nich vytváří profily znázorněné výše. První implementovaná komponenta aplikace tento textový výstup převádí do binární reprezentace (resp. do dvourozměrného bytového pole o rozměru n x 20 položek). Kromě toho extrahuje pozici hranice zapsanou na prvním řádku FASTA souboru umístěnou hned za identifikátorem sekvence a provádí konverzi názvu souboru podle konvence zobrazené na spodním obrázku.

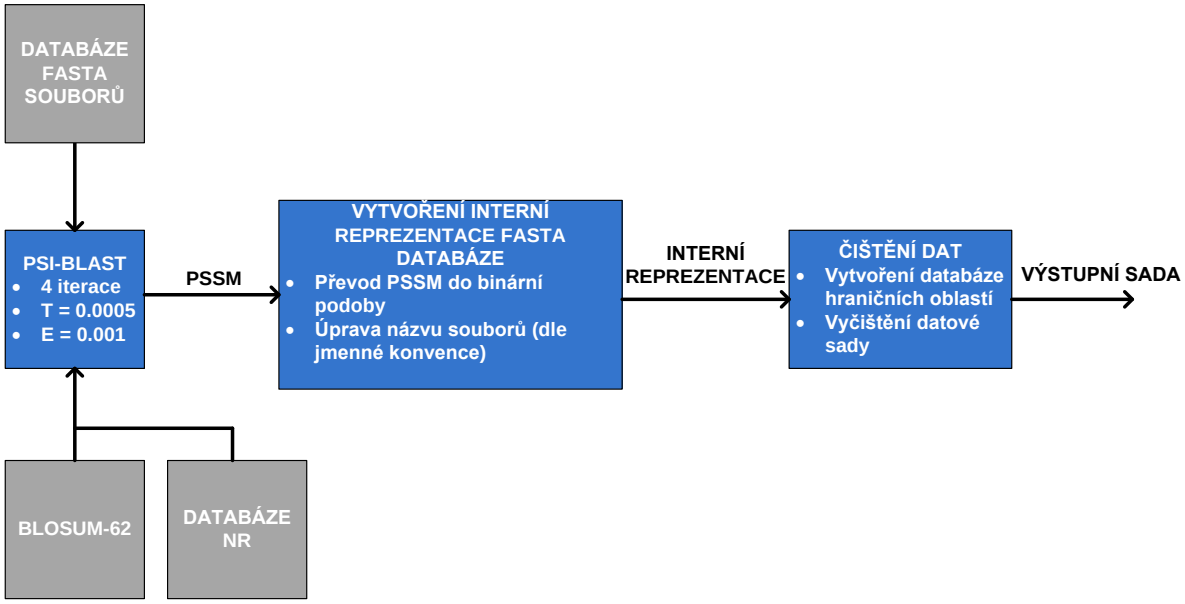


Obrázek 5.8: Jmenná konvence umožňující identifikovat hranici domén z názvu trénovacího souboru.

Právě zmíněná konvence je poměrně důležitá, protože její dodržení umožňuje během procesu trénování klasifikátoru i jeho následného testování řádně identifikovat tu část sekvence, která náleží do oblasti hranice. Sada dat uložená v interní reprezentaci navíc výrazně urychluje oba procesy (generování profilu z vložené FASTA sekvence zabírá převážnou část délky výpočtu implementovaného algoritmu).

Hned po převodu tréninkových dat do interní reprezentace je provedeno jejich čištění. Důvodem tohoto kroku je možná přítomnost hraničních vzorů v nehraničních oblastech PSSM, která by znemožňovala kvalitní proces učení neuronové sítě. Čistění jsem implementoval v podobě speciálního nástroje, jehož použití je blíže popsáno v odpovídající příloze a který provádí svůj výpočet ve dvou krocích.

1. Nejprve jsou z tréninkových profilů extrahovány vzory z hraničních oblastí.
2. Iterativně jsou kontrolovány všechny uložené PSSM. Pokud se nějaká jejich nehraniční část shoduje s nalezenou hraniční oblastí, je jeden náhodný řádek profilu z dané oblasti vynulován. Před vynulováním řádku je samozřejmě nejprve zkontrolováno, jestli nezasahuje do příslušné hranice.



Obrázek 5.9: Neformální diagram znázorňující jednotlivé fáze převodu a čištění tréninkové sady dat tak, jak je popsáno výše v textu.

Aby byl popis přípravy dat kompletní, zbývá ještě vysvětlit, jak se odlišuje chování dříve popsaných metod v režimu predikce nových sekvencí. Analyzovaná sekvence je znovu nejprve přivedena na vstup programu PSI-BLAST, který vytvoří PSSM. Profil je následně převeden do vnitřní reprezentace, tentokrát bez dodržení jmenné konvence pro název souboru (ta není algoritmem predikce pochopitelně využívána) a tím úpravy končí.

Poznámky k paralelizaci realizované implementace

Při návrhu a realizaci obou právě zmíněných nástrojů jsem se snažil jejich části vhodně paralelizovat. Výpočet PSSM a následný převod do vnitřní reprezentace proto probíhá souběžně pro *X* vstupních FASTA souborů, kde je *X* rovno počtu dostupných procesorů (pozn. nemusí přímo odpovídat počtu fyzických procesorů). Obě fáze čistění dat jsem také paralelizoval, a to s využitím knihovny .net TPL (Task Parallel Library), která obsahuje podporu paralelního modelu výpočtu zaměřeného na tzv. úlohy (tasks). Opět jsou tak hraniční vzory souběžně vyhledávány a poté kontrolovány s využitím všech dostupných výpočetních prostředků. Konkrétní implementované paralelní postupy jsou přehledně zdokumentovány v přiložených zdrojových kódech.

Vytvoření vstupního vektoru neuronové sítě

Během učení i následné predikce pracuje použitá neuronová síť se vstupními vektory sestavenými na základě získaných PSSM. Každý vektor se skládá ze 400 hodnot normalizovaných do intervalu <-1,1> aplikací sigmoidální funkce (6) na původní normalizované frekvence *x*. Vektor je ve zpracovávané sekvenci aminokyselin (reprezentované pomocí PSSM) konstruován pro každé residuum na pozici *i* v rozsahu *(10, délka sekvence -10)* tak, že jsou řádky PSSM v rozmezí *i – 10 ... i + 10* postupně za sebou vkládány do jediného pole o výsledných 400 položkách. Ke zkrácení sekvence je přistoupeno, aby se předcházelo případným nežádoucím vlivům nuceného doplnění "umělých" hodnot do vstupních vektorů residuí mimo popsaný rozsah.

$$x' = \frac{1}{1 + e^{x}} \tag{6}$$

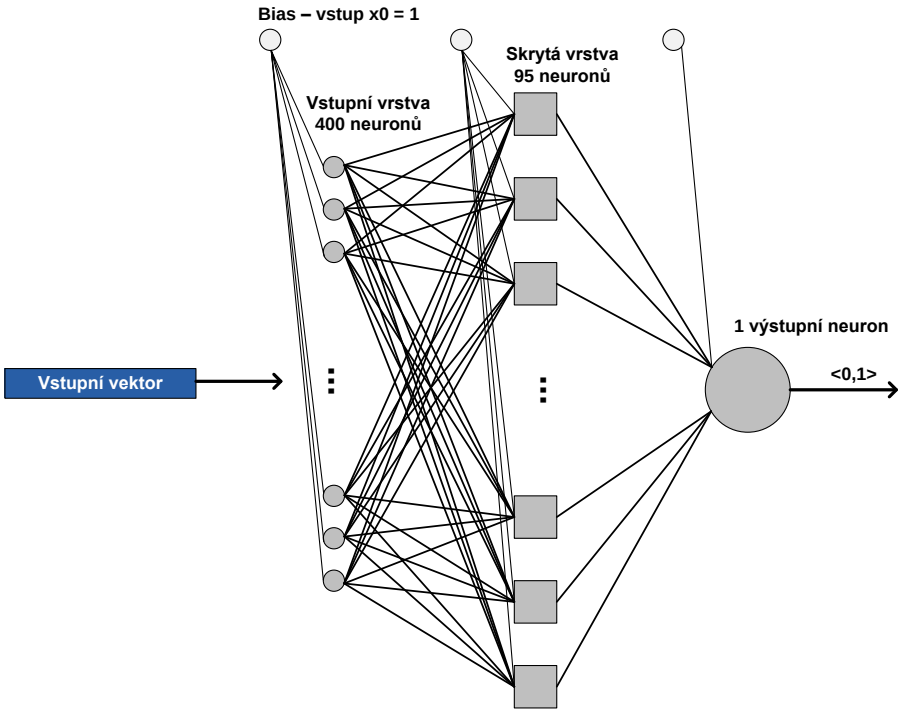
5.4 Neuronová síť

Závěrečnou část kapitoly jsem se rozhodl věnovat použité metodě strojového učení, a to neuronové síti. Dále popsaná neuronová síť představuje jednu z nejdůležitějších a možná také nejkomplexnějších součástí celého realizovaného postupu. Při návrhu její topologie jsem vycházel z výsledků popsaných v [18], konkrétní implementaci jsem ale založil na osobních zkušenostech a studijním materiálu [27]. Během realizace algoritmu a následného testování jsem pod vlivem nečekaných komplikací síť vylepšoval v různých aspektech, kterými se budu dále zabývat.

5.4.1 Obecné vlastnosti realizované neuronové sítě

Aplikovaná metoda využívá dopřednou neuronovou síť, jejíž váhy jsou optimalizovány algoritmem zpětného šíření chyby (Back-Propagation). Rozměry sítě jsem převzal z [18] a sestávají ze 400 neuronů ve vstupní vrstvě, 95 neuronů v jediné skryté vrstvě a jediného výstupního neuronu. Síť je

využita jako binární klasifikátor, který svým výstupem v rozsahu <0, 1> určuje, jestli zadaný vstupní vektor představuje hraniční oblast proteinu (pro hodnotu 1).



Obrázek 5.10: Architektura použité dopředné neuronové sítě s jedinou skrytou vrstvou o rozměru 95 neuronů. Váhy sítě, stejně jako váhy vstupů x_0 jsou optimalizovány algoritmem Back-Propagation.

V rámci neuronové sítě je použita běžná lineární bázová funkce (7) s vektorem vstupů $\vec{x}$ a vektorem vah $\vec{w}$. Aktivační funkce (8) je sigmoidální, parametr λ jsem experimentálně nastavil na hodnotu 0,8. Výpočet sítě jsem se rozhodl urychlit pomocí paralelizace, implementovaný algoritmus postupuje po vrstvách. V rámci vstupní a skryté vrstvy jsou neurony rozděleny do X souvislých skupin, kde X odpovídá počtu dostupných logických jader konkrétního procesoru. Všechny skupiny jsou následně zpracovány paralelně – výstupy neuronů v rámci skupiny jsou vypočítávány sekvenčně. Samotná podstata dopředné sítě (především propojení neuronů) pak umožňuje tento postup provést velmi efektivně, protože není nutné řídit přístup ke sdílené paměti. Všechny souběžné úlohy totiž zapisují do vlastního separátního regionu paměti a sdílené čtení hodnot zapsaných předchozí vrstvou nepředstavuje z pohledu synchronizace žádný problém (synchronizaci vyžaduje pouze přechod na další vrstvu sítě, kdy musí být nejprve dokončen probíhající výpočet všech skupin neuronů). Výše popsaný postup je kompletně zdokumentován v přiložených zdrojových kódech.

$$u = \sum_{i=1}^{n} w_i x_i \tag{7}$$

$$y = \frac{1}{1 + e^{(-\lambda u)}} \tag{8}$$

5.4.2 Realizovaná podoba metody zpětného šíření chyby

Použitý algoritmus učení sítě vychází z obvyklé implementace Back-Propagation, přidává však několik speciálních rozšíření, která budu postupně rozvádět. Základem metody zpětného šíření chyby je úprava vah podle vztahu (9).

$$\Delta^l w_{ji} = \mu^l \delta_j^l x_i \tag{9}$$

Předchozí vztah popisuje úpravu váhy *i-tého* vstupu *j-tého* neuronu ve vrstvě *l*. Parametr μ představuje koeficient učení, ${}^l x_i$ značí hodnotu na *i-tém* vstupu neuronu. Zbylý parametr ${}^l \delta_j$ představuje zpětně propagovanou chybu, respektive vliv, který na ni má právě zpracovávaný neuron (což popisuje následující vztah (10)).

$${}^l\delta_j = -\frac{\partial E}{\partial {}^l u_j} = -\frac{\partial E}{\partial {}^l y_j}\frac{\partial {}^l y_j}{\partial {}^l u_j} \tag{10}$$

Ve vztahu (10) *E* značí chybu sítě dále popsanou rovnicí (11). Parciální derivace sigmoidální aktivační funkce podle hodnoty bázové funkce je rovna vztahu (12). Parciální derivace chyby sítě podle hodnoty výstupu aktivační funkce se vypočte derivací vztahu (11). Konečný tvar vztahu (10) pro výstupní vrstvu znázorňuje (13) a pro zbylé vrstvy (14).

$$E = \frac{1}{2}\sum_{j=1}^{n_L}(d_j - {}^L y_j)^2 \tag{11}$$

Výpočet chyby sítě (11) obsahuje několik dalších proměnných, n_L určuje počet neuronů ve výstupní vrstvě, d_j požadovaný výstup *j-tého* neuronu výstupní vrstvy a ${}^L y_j$ aktuální výstup *j-tého* neuronu výstupní vrstvy.

$$\frac{\partial {}^l y_j}{\partial {}^l u_j} = \lambda {}^l y_j (1 - {}^l y_j) \tag{12}$$

$${}^L\delta_j = (d_j - {}^L y_j)\lambda {}^L y_j (1 - {}^L y_j) \tag{13}$$

$$^{l-1}\delta_j = \sum_{k=1}^{n_l} \left({}^{l}\delta_k \, {}^{l}w_{kj} \right) \lambda \, {}^{l-1}y_j (1 - {}^{l-1}y_j) \tag{14}$$

V předchozí rovnici se opět objevuje nový parametr n_l, který představuje počet neuronů v aktuálně zpracovávané vrstvě l. Výše rozepsaný postup úpravy vah sítě jsem v implementované metodě doplnil o dvě celkem běžná rozšíření. Prvním z nich je výpočet proměnlivého koeficientu učení μ, který klesá s postupně rostoucím počtem iterací metody a je dán vztahem (15). Hodnotu tohoto koeficientu jsem dle doporučení v [27] ošetřil tak, aby vždy náležela do intervalu <0,1, 0,9>.

$$\mu = \frac{\mu_0}{(1 + \frac{k}{K})} \tag{15}$$

Vztah pro výpočet koeficientu učení (15) obsahuje několik proměnných a konstant. Proměnná k představuje aktuální krok metody, hodnotu konstanty K jsem experimentálně stanovil (vzhledem k průměrnému celkovému počtu iterací) na 1700 a hodnotu konstanty μ_0, představující výchozí koeficient učení, na 0,25.

Druhým poměrně standardním rozšířením metody Back-Propagation je použití tzv. α momentu, který částečně kompenzuje záchvěvy sítě během učení. Výpočet změn vah sítě (9) je upraven tak, aby nevycházel pouze z aktuálního gradientu, ale bral v potaz také jejich předchozí změnu, což zachycuje rovnice (16), která při stanovení aktuálních vah *w(k+1)* využívá také váhy předchozí $w(k)$. Jako použitou hodnotu konstanty α jsem experimentálně zvolil 0,5.

$$\Delta^l w_{ji}(k+1) = \mu^l \delta_j^l x_i(k) + \alpha \Delta^l w_{ji}(k) \tag{16}$$

Pro trénování neuronové sítě jsem realizoval ještě jedno doporučení specifikované v [27], kterým je náhodný výběr trénovacích vzorů. V mém případě je tato funkcionalita realizována tak, že jsou v jedné iteraci metody vždy zpracovány všechny soubory trénovací sady dat (v interní reprezentaci) v náhodném pořadí. Obsah každého konkrétního souboru je však zpracován sekvenčně.

Pseudokód realizované metody Back-Propagation

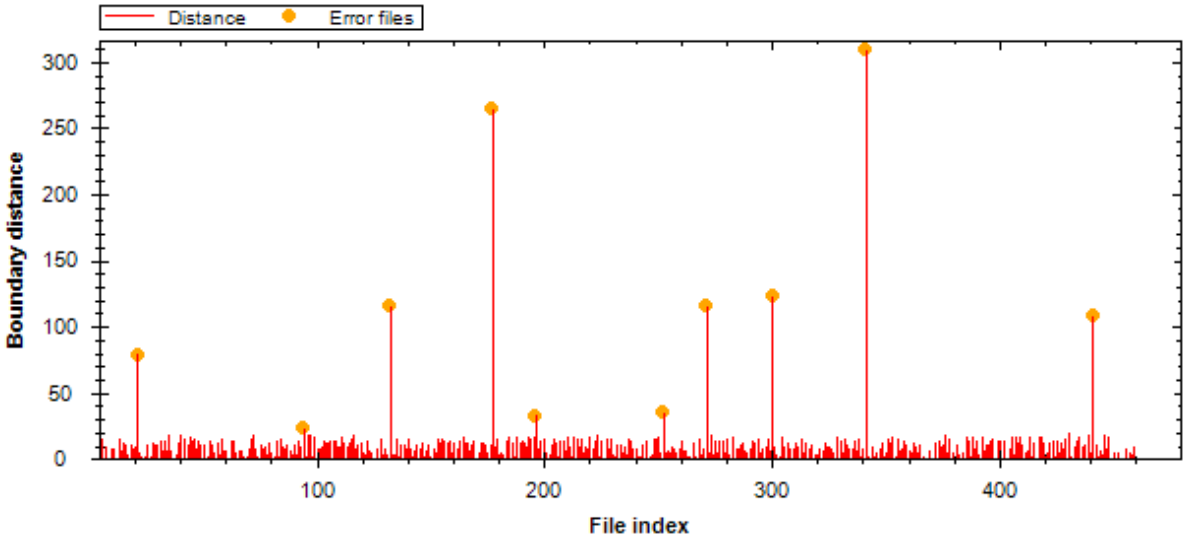
Na následující straně je předveden pseudokód metody zpětného šíření chyby, jehož značná část kopíruje principy teoreticky popsané výše. Algoritmus zastaví, pokud je globální chyba sítě nižší nebo rovna požadované, nebo bylo dosaženo maximálního možného počtu iterací. Jednou iterací metody se zde rozumí průchod celé trénovací sady. V rámci kódu bych chtěl upozornit především na místa znázorněná červeně, která představují úpravy základního algoritmu BP.

```
// Inicializace
Aplikuj genetický algoritmus pro inicializaci vah sítě
Inicializace počtu provedených iterací k = 0
Vytvoř zásobník History_stack
do{
        globální_chyba = 0
        do
        {
            // Odezva sítě
           Načti tréninkový vzor a požadovanou odezvu sítě
           Vypočítej odezvu sítě
           Vypočítej lokální chybu sítě localError dle (11)
           Globální_chyba += 0.5 * (localError)^2

            // Výpočet nových vah
           Výpočet ^l δ_j podle (13) a (14)
           Výpočet vah sítě podle (16)

        } while Není zpracována cela tréninková sada;

        Změna koeficientu učení μ podle (15)
        History_stack.Push(Globální_chyba)

        // Specifická rozšíření
        if (k % kontrolní_počet_iteracíI = 0 and
            Re-validace sítě indikuje přeučení)
            return;

        if (k % kontrolní_počet_iteracíII == 0 analýza
            History_stacku indikuje zákmity sítě)
        {
            If (analýza History_stacku indikuje zákmity sítě)
            {
                  Aplikuj genetický algoritmus
            }
            Vyprázdnění zásobníku History_stack
        }

        Inkrementuj počet kroků (k);
    } while (globální_chyba > minimální_chyba and k < max_k) ;
```

Prvním místem, které si zaslouží pozornost, je re-validace výstupu sítě. V rámci tohoto kroku je stav sítě otestován na zvoleném re-validační množině proteinových sekvencí, která není obsažena v trénovací sadě. Cílem je ověřit schopnost sítě zobecňovat s právě počítanými vahami. Princip zmíněné kontroly je velmi jednoduchý – porovná se výsledek predikce sítě v aktuálním stavu s výsledkem předchozím, a pokud dojde k poklesu úspěšnosti je výpočet BP okamžitě zastaven. Abych snížil dodatečnou výpočetní zátěž tohoto postupu, neprovádím ho v každém kroku (v implementované verzi pouze každou pátou iteraci).

Právě popsaná úprava algoritmu BP se ukázala jako velmi důležitá. Během testování totiž síť projevovala tendenci se poměrně rychle "přeučovat", což bez tohoto zásahu mělo pochopitelně výrazný dopad na kvalitu dalšího zobecnění.

Posledním vylepšením je použití genetického algoritmu, k tomuto kroku jsem opět přistoupil na základě prvotního testování metody BP, která se na poměrně rozsáhlé trénovací sadě (500 proteinů a přibližně 15000 vstupních vektorů) učila nepříjemně pomalu – z pohledu poklesu globální chyby sítě vzhledem k počtu iterací. Po bližším zkoumání tohoto problému jsem zjistil, že zmíněné chování způsobují vzory trénovací sady, které sice nepatří do hraničních oblastí, ale jsou jim nápadně podobné (nikoliv identické). Procedura BP pod vlivem těchto vzorů často oscilovala kolem několika málo řešení, která střídavě vylepšovala chování sítě pro zbylé, nebo problémové vzory (tento problém dokumentuje obrázek 5.11). Abych umožnil uvolnění algoritmu BP z tohoto pomyslného lokálního extrému, rozhodl jsem se zavést do výpočtu vah náhodný prvek. Při výběru různých alternativ, jak tento postup realizovat, jsem se nakonec rozhodl pro genetický algoritmus, který kromě toho, že do výpočtu vnáší požadovaný náhodný aspekt, tak ho také vnáší velmi efektivním způsobem (současně váhy optimalizuje).



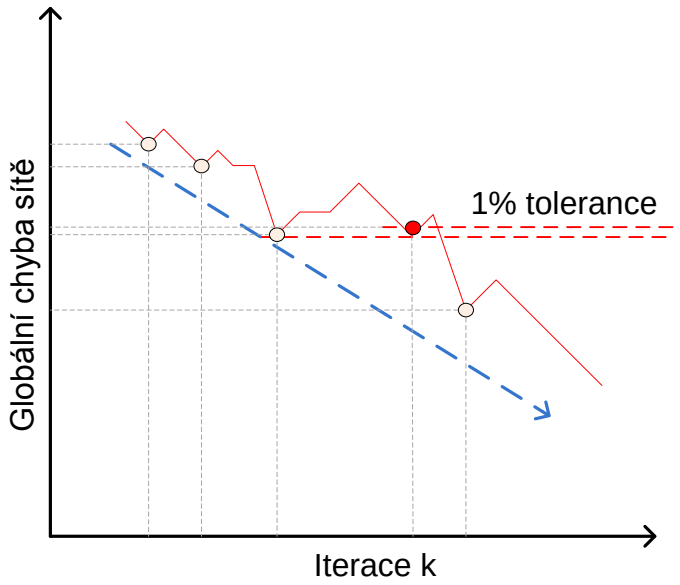
Obrázek 5.11: Graf znázorňující vzdálenosti predikovaných hranic od reálných hranic domén (vertikální osa) pro všechny sekvence v rámci tréninkové sady dat (horizontální osa). Na obrázku je patrných několik problémových řetězců, které mohou obsahovat oblasti podobné hranicím domén v nehraničních částech.

Vytvořený genetický algoritmus používám v rámci učení neuronové sítě na dvou místech. Prvním z nich je inicializace vstupních vah a druhým výše popsané řešení oscilace. Bližší detaily k jeho implementaci poskytuje následující část kapitoly. Běh algoritmu je poměrně výpočetně náročný, naštěstí se ukázalo - že už velmi malý počet individuí i generací vede k požadovanému výsledku (reálně je použito pouze 10 jedinců v 10 generacích). Rychlost učení sítě se nasazením tohoto vylepšení výrazně zvýšila (na místo původních více než 3000 iterací - nyní úplně postačuje méně než 1000 pro dosažení obdobného výsledku). Jako pozitivní se také ukázalo chování generovaných vah, které obvykle nepůsobí velké výkyvy při následném běhu BP (globální chyba sítě

buď přímo klesne, nebo mírně vzroste, ale ve většině případů ne o víc než 2% její hodnoty před úpravou).

Důležitou součástí popsaného vylepšení je samotná detekce oscilace metody BP, kterou provádím následujícím způsobem. V každé iteraci metody zaznamenávám globální chybu sítě, pokud je zaznamenaných chyb dostatečný počet (v implementaci využívám 20 iterací), provádím kontrolu trendu chyby. Postupuji od nestarší hodnoty po nejmladší a vyhledávám lokální minima. Následně provádím kontrolu nalezených minim, která by v ideálním příkladě měla tvořit klesající posloupnost. Reálně však v rámci sítě dochází téměř vždy k určitým zákmitům. Proto procházím nalezená lokální minima a kontroluji platnost vztahu (17), kde $E_L(\mathrm{k})$ značí nalezenou lokální chybu v příslušné iteraci k. Pokud je podmínka splněna (0,99 představuje 1% toleranci růstu) pro všechna lokální minima, považuji proces učení neuronové sítě jako bezproblémový, historii globálních chyb vyprázdním a celý cyklus se opakuje. V opačném případě historii globálních chyb opět vyprázdním, ale následně aktivuji výpočet genetického algoritmu. Obrázek 5.12 dokumentuje právě popsaný postup.

$$\boldsymbol{E_L(k-1) \leq E_L(k) * 0,99} \tag{17}$$

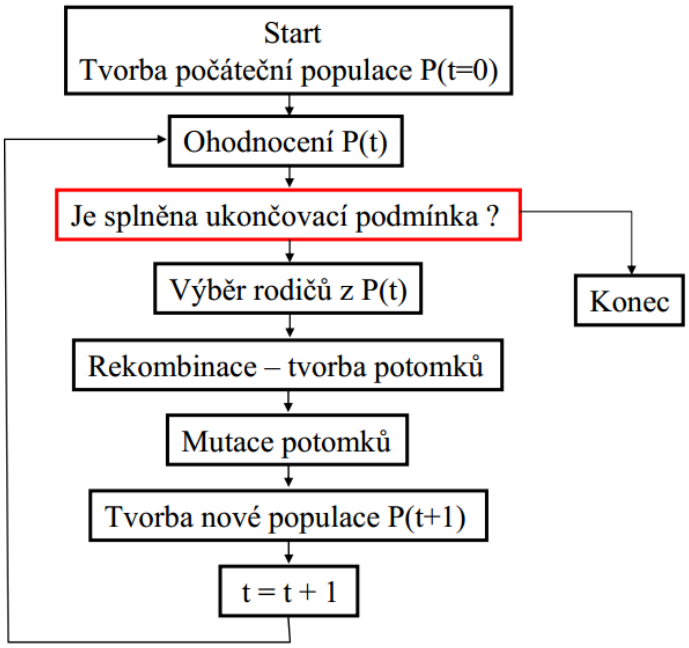


Obrázek 5.12: Znázornění mechanismu detekování oscilace metody Back-Propagation, modře je znázorněn ideální průběh lokálních extrémů (vyznačené body) průběhu globální chyby sítě. Reálný průběh chyby vykazuje mírné zákmity, jejichž hodnota je však v popsané 1% toleranci. Celkový průběh tedy vykazuje známky poklesu, proto není nutné použít genetický algoritmus.

5.4.3 Použitý genetický algoritmus

Předchozí část textu se zabývala metodou Back-Propagation a různými aspekty jejího vylepšení, mezi jeden z nejdůležitějších patří detekce uváznutí v lokálním extrému a následné řešení této situace užitím genetického algoritmu, na který se nyní zaměřím.

Při realizaci genetického algoritmu jsem vycházel z osobních zkušeností a studijního textu [28]. Implementoval jsem základní verzi algoritmu, kterou znázorňuje model na obrázku 5.13.



Obrázek 5.13: Model implementovaného genetického algoritmu převzatého z [28].

Jedinec populace reprezentuje vektor vah neuronové sítě o rozměru 38095 reálných hodnot (typu float). Tvorba počáteční populace sestává z vytvoření náhodných vektorů, a v případě řešení oscilace, také ze začlenění původního vektoru vah neuronové sítě.

Ohodnocení populace je realizováno testováním každého jedince (resp. vektoru vah), kdy je stejným způsobem, jako u algoritmu BP vypočítána globální chyba příslušné sítě, která je následně převedena na hodnotu fitness podle vztahu (18).

$$\boldsymbol{Fitness\ jedince = 10000 - globální\ chyba\ sítě} \tag{18}$$

Dalším krokem je výběr rodičů, který je realizován tzv. turnajovou selekcí. Během které je náhodně zvoleno *n* jedinců (n se v realizovaném algoritmu rovná třetině populace) původní populace. Z těchto jedinců je vybrána dvojice rodičů s nejlepší fitness hodnotou, se kterými se následně provede rekombinace.

Rekombinaci neboli tvorbu potomstva jsem prováděl podle vztahu (19) pro geny s reálnými hodnotami, který jsem převzal z [28]. V následujícím vztahu představuje $Var_i^{O_j}$ hodnotu genu i potomka j, a_i je náhodná hodnota pro gen i z intervalu <-0,25, 1,25>. $Var_i^{P_j}$ značí hodnotu genu i rodiče j.

$$\boldsymbol{Var_i^{O_1} = Var_i^{P_1} a_i + Var_i^{P_2}(1 - a_i)}$$
$$\boldsymbol{Var_i^{O_2} = Var_i^{P_1}(1 - a_i) + Var_i^{P_2}(a_i)} \quad (19)$$

Mutaci provádím s pravděpodobností 0,001 výběrem náhodného potomka a jeho náhodného genu a následným přičtením malé náhodné hodnoty (z rozsahu 0-0,1) k vybranému genu. Vytvořené potomky je nutné ohodnotit podle výše popsaného principu.

Na předchozí kroky navazuje obnova populace, kterou jsem realizoval na základě tzv. elitismu, kdy jsou v mojí implementaci obě populace (jak původní jedinci, tak právě vytvoření potomci) seřazeny podle fitness funkce a z rodičovské populace je třetina nejhorších jedinců nahrazována kvalitnějšími zástupci z nejlepších potomků.

Na závěr každé iterace metody vybírám jedince populace, který je reportován na výstupu a je jím buď nejlepší individuum v populaci (pokud se nejedná o původní vektor vah neuronové sítě), nebo druhý nejlepší jedinec – pochopitelně z hlediska fitness funkce. Metoda pokračuje podle modelu na obrázku 5.13, dokud není dosaženo požadovaného počtu iterací.

Paralelizace metody

Abych urychlil výpočet právě popsaného algoritmu, opět jsem se pokusil některé jeho části vhodným způsobem paralelizovat. Vhodná místa paralelizace (tzv. hotspoty) jsem identifikoval v rámci ohodnocování populace a tvorbě potomstva. Tyto procesy představují nesporně nejvyšší výpočetní zátěž celého postupu.

Během ohodnocování populace jsou jedinci rozděleni do *X* souběžně zpracovávaných skupin (*X* odpovídá počtu logických jader procesoru). Zástupci v rámci každé skupiny jsou ohodnoceny popsaným způsobem sekvenčně.

Druhé popsané místo paralelizace je zpracováváno podobným způsobem. Vytváření požadovaného počtu potomků je opět rozděleno mezi *X* procesů (resp. úloh), které probíhají paralelně. V rámci každé úlohy je sekvenčně opakován cyklus kroků vedoucí ke generování potomka obsahující výběr rodičů, křížení a mutaci.

Obě předchozí místa skýtají významnou výhodu – není v nich třeba řídit přístup ke sdílené paměti. Souběžné operace zapisují své výsledky do separátních regionů paměti a souběžné čtení paměťových míst (při generování potomků) nepředstavuje problém z hlediska nutnosti synchronizace. Pro bližší informace o implementovaném postupu odkazuji čtenáře na příslušný zdrojový kód.

5.5 Integrace regulárních výrazů z databáze PROSITE

Na úplný závěr kapitoly se pokusím vyjasnit použití databáze PROSITE v rámci implementované metody. Zmíněná databáze je přístupná ve formě webového serveru z adresy http://prosite.expasy.org/, její zajímavou vlastností je, že obsahuje definice některých známých domén ve formě regulárních výrazů. Navíc je koncipována jako jediný textový soubor, který je uživatelsky snadno interpretovatelný.

V rámci implementovaných nástrojů používám regulární výrazy z této databáze, bezprostředně po predikci proteinových domén (výše popsanou metodou) k rozšíření uživatelského výstupu o další analýzu vstupní sekvence (což bude předvedeno v následující kapitole).

Anotaci vstupní sekvence provádím velmi primitivním způsobem, kdy nad ní vyhledávám jednotlivé vzory domén reprezentované regulárními výrazy. Abych tuto činnost alespoň mírně urychlil, opět jsem využil paralelizaci. Regulární výrazy jsem rozdělil do X skupin, které jsou zpracovávány paralelně, v rámci každé skupiny jsou ve vstupním řetězci sekvenčně vyhledávány vzory odpovídající jednotlivým regulárním výrazům a informace o nalezených shodách jsou ukládány do sdílené kolekce. Nevýhodou tohoto přístupu je nutnost synchronizace zápisu do zmíněné kolekce. Realizovaný postup je zdokumentován v příslušném zdrojovém kódu.

6 Dosažené výsledky a námět pro další rozvoj

V předposlední kapitole práce navazuji na části předchozí a kladu si za cíl nejprve předvést realizované nástroje, z nichž se soustředím především na ty s implementovaným uživatelským rozhraním. Popis funkce a ovládání zbylých programů bude ponechán v příslušné sekci příloh.

Následně vysvětlím metodiku testování právě definovaného algoritmu predikce hranic proteinových domén včetně způsobu hodnocení úspěšnosti jeho výstupů, a také původu a složení testovací sady dat.

Zbytek práce pak bude obsahovat výsledky dosažené během testování spolu s několika grafovými výstupy hodnotícími vybrané testované aspekty. Na závěr uvedu možné varianty dalšího rozvoje implementované metody.

6.1 Realizované nástroje

Nejprve se pokusím zaměřit, tak jak bylo avizováno na začátku kapitoly, na vytvořené nástroje, ty jsem se zde rozhodl rozdělit do dvou skupin, a to na podpůrné a výkonné.

První z nich slouží především k převodu vstupních dat z původní sekvence aminokyselin ve formátu FASTA do interní reprezentace založené na PSSM vytvořené metodou PSI-BLAST a k následnému trénování klasifikátoru (resp. použité neuronové sítě). Zmíněný druh nástrojů jsem se rozhodl implementovat formou konzolových aplikací, a to bez přítomnosti grafického uživatelského rozhraní, protože jejich očekávaná funkcionalita by měla být co nejvíce automatizovaná a vyžadovat tak naprosté minimum uživatelských zásahů. Obvykle jsou vyžadovány pouze volby konkrétních parametrů daných metod ve formě startovacích argumentů. O průběhu výpočtu je uživatel stručně informován a je mu obvykle umožněno jeho okamžité přerušení (resp. následné obnovení).

Druhá skupina vytvořených aplikací, provádí výpočet, jehož výsledky je nutné prezentovat uživateli v rozumné podobě, proto je u nich implementováno příslušné grafické uživatelské rozhraní. Kromě toho je většinou vyžadována hlubší uživatelská interakce, protože tento typ nástrojů má spíše integrační charakter. Následující seznam stručně prezentuje funkci a rozdělení jednotlivých vytvořených programů (názvy odpovídají reálné implementaci), bližší informace o jejich nastavení a běhu obsahují příslušné části sekce příloh.

1. Podpůrné metody
    a. *TrainingSetConvertor* je nástroj realizující kompletní převod vstupní sekvence/sekvencí do dříve popsané interní reprezentace s využitím nástroje PSI-BLAST.

b. *DataCleaning* je nástroj provádějící výše popsané čistění dat doplněné o kontrolu přítomnosti celé hranice v konkrétní sekvenci.
c. *PrositeSupport* je nástroj, který ze souboru reprezentujícího databázi PROSITE vybírá regulární výrazy domén, převádí je do tvaru podporovaného zvoleným implementačním prostředím a ukládá je do interní reprezentace.
d. *TrainingSetAnalyzer* je nástroj, který s využitím zpracované interní reprezentace domén z databáze PROSITE provádí analýzu trénovací / testovací sady dat z pohledu přítomnosti jednotlivých uložených vzorů.
e. *TrainingTool* je jeden z nejdůležitějších nástrojů provádějící učení implementované neuronové sítě.

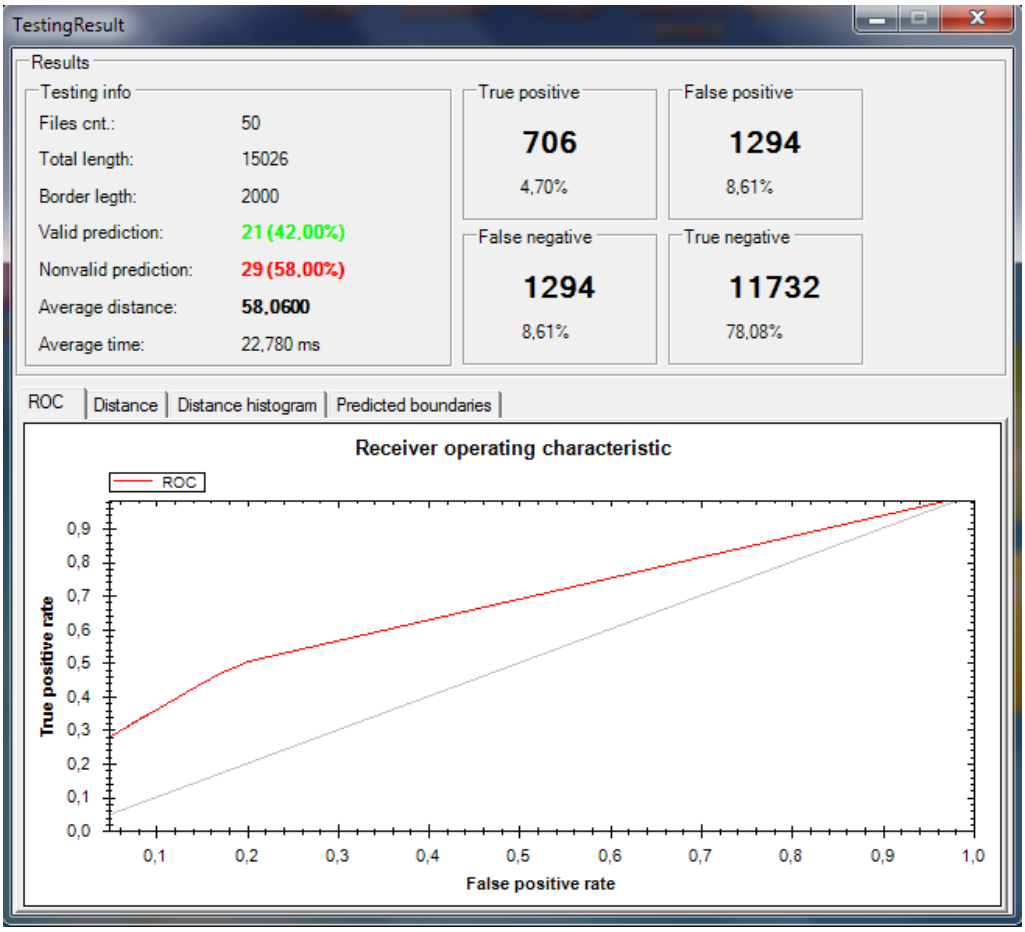
2. Výkonné metody
   a. *BPTesting* je konzolová aplikace doplněná o grafický uživatelský výstup, která umožňuje testování vypočtených vah neuronové sítě a vizuální prezentaci dosažených výsledků.
   b. *SimplePresenter* je aplikace kompletně založená na uživatelském rozhraní umožňující uživateli analyzovat konkrétní proteinové sekvence v několika režimech.

6.1.1 BPTesting

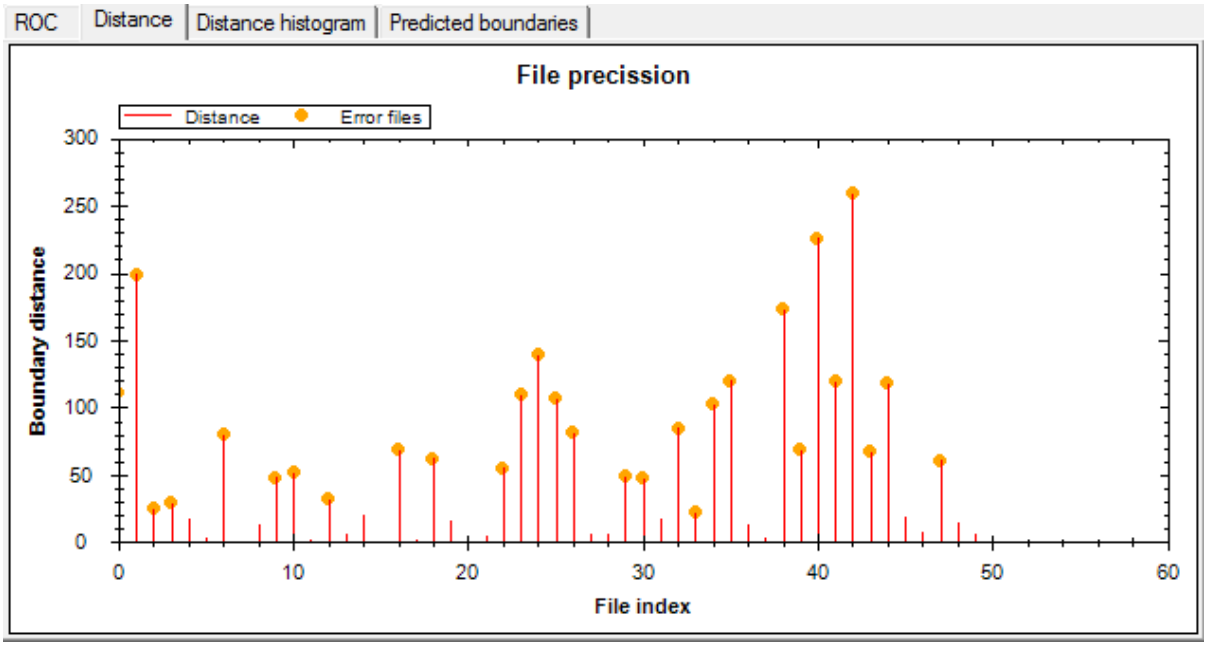
První prezentovaný nástroj, jak bylo řečeno již v předchozím rozdělení, provádí testování naučených vah neuronové sítě, které reprezentují binární klasifikátor. Na svém vstupu předpokládá vypočítané váhy (resp. soubor s jejich reprezentací), cestu ke složce obsahující testovací sadu souborů v interní reprezentaci (testovací data budou dále popsána v následujícím textu).

Nástroj následně provádí predikce hranic domén pro jednotlivé testovací soubory s využitím předaného nastavení neuronové sítě a výsledky přitom zaznamenává. Uživatel je o průběhu testování informován pomocí výpisů do konzole. Po dokončení testovacího cyklu jsou uživateli prezentovány následující informace v uživatelském rozhraní zobrazeném na následujících obrázcích.

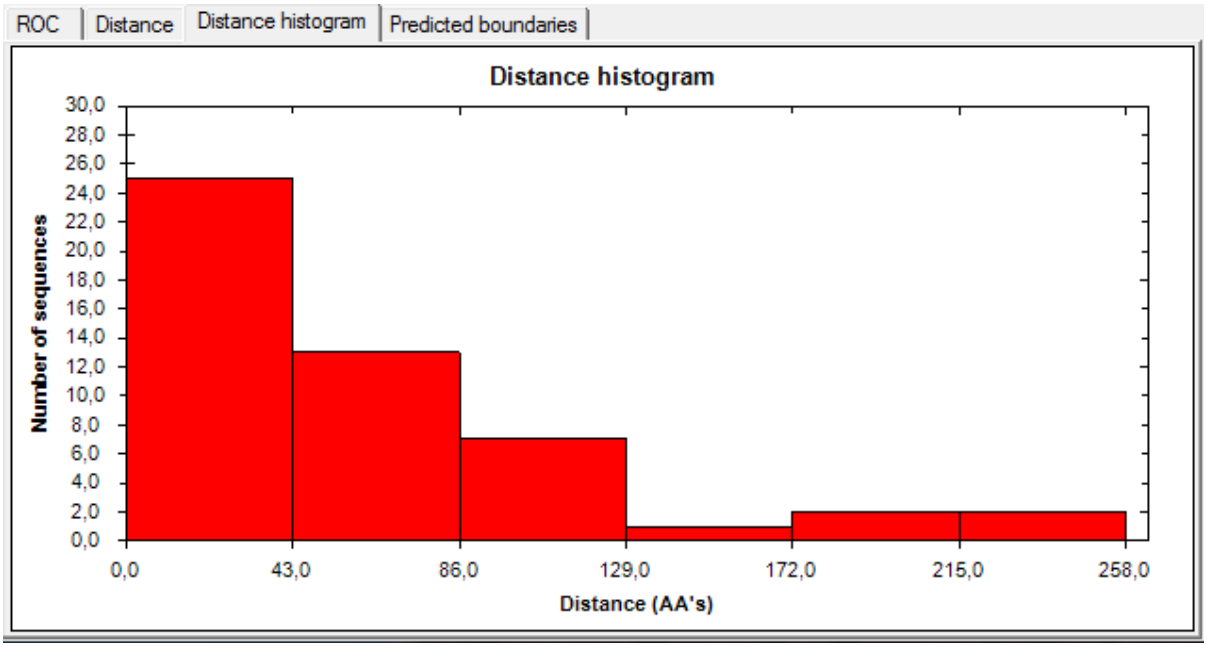
- Stav překryvu nalezené hranice s reálnou hranicí pro jednotlivé testované soubory.
- ROC křivka (její výpočet je popsán na konci této části textu).
- Graf vzdáleností predikovaných hranic od hranic reálných - pro všechny soubory.
- Histogram vzdáleností predikované hranice od reálné (výpočet bude opět naznačen na konci této části textu).
- Průměrná vzdálenost predikovaných hranic od reálných.
- Hodnoty TP, FP, TN a FN vypsané v počtech odpovídajících residuí.
- Počet a procento správně / nesprávně predikovaných souborů.
- Celková délka zpracovaných sekvencí a délka hraničních oblastí v počtu residuí.



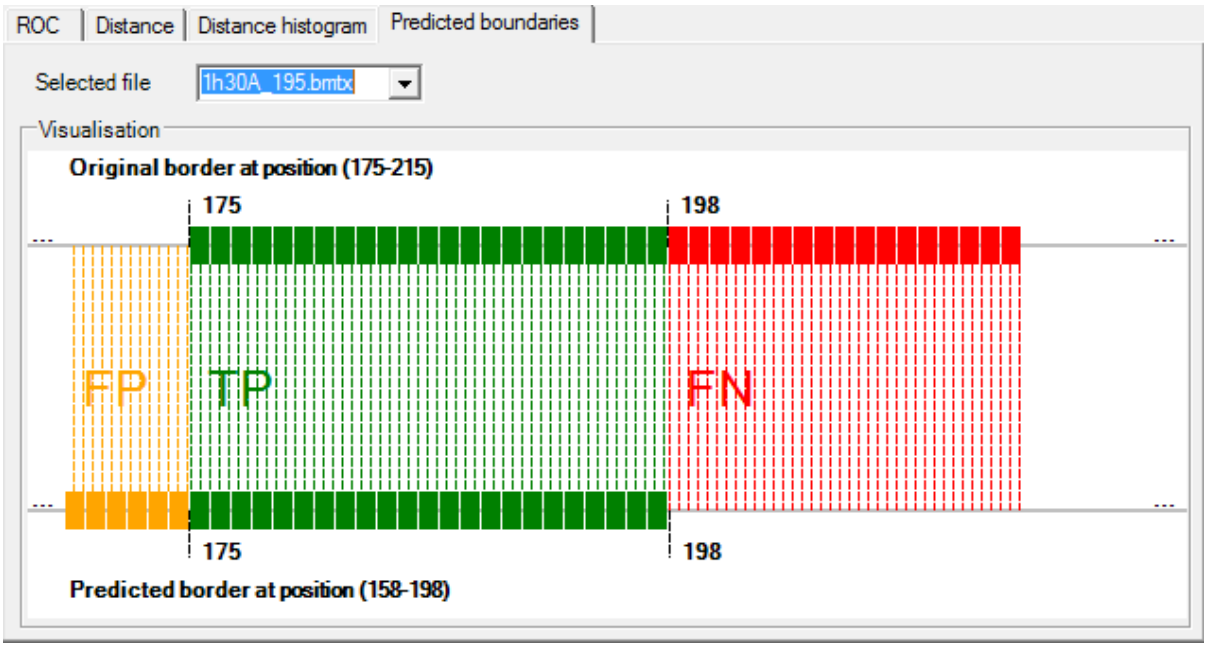
Obrázek 6.1: Uživatelské rozhraní aplikace testující výsledky naučené neuronové sítě. Ve spodní části je graf ROC.



Obrázek 6.2: Sloupcový graf vzdáleností predikovaných hraničních residuí od pozic reálných hranic pro jednotlivé soubory testovací sady. Kroužkem jsou označeny soubory, u kterých došlo k nesprávné predikci.



Obrázek 6.3: Histogram vzdáleností predikovaných hranic od hranic reálných.



Obrázek 6.4: Vizualizace překryvu predikované a skutečné hranice pro jednotlivé testovací soubory.

ROC křivka

Výpočet ROC křivky jsem převzal z materiálu [29] a je založen na vztahu senzitivity (vertikální osa zobrazující relativní četnost TP) zapsaná vztahem (20), ku specificitě (horizontální osa zobrazující relativní četnost FP) dané vztahem (21). Použité pojmy TP, FP, TN a FN budou v kontextu implementovaného testování vysvětleny v kapitole zabývající se metodikou testování.

$$\boldsymbol{TPR} = \frac{\boldsymbol{TP}}{\boldsymbol{TP} + \boldsymbol{FN}} \tag{20}$$

$$\boldsymbol{FPR} = \frac{\boldsymbol{FP}}{\boldsymbol{FP} + \boldsymbol{TN}} \tag{21}$$

Histogram vzdáleností

Výpočet histogramu jsem opět provedl standardním způsobem naznačeným v materiálu [30]. Pro každý zpracovaný soubor jsem zaznamenal vzdálenost predikované hranice od reálné. Po zaznamenání všech těchto vzdáleností jsem určil interval, ve kterém leží jejich hodnoty. Zjištěný interval jsem následně rozdělil mezi určité množství tzv. košů (resp. podintervalů) - dané vztahem (22). Poté stačilo pouze určit počty hodnot v jednotlivých koších.

$$\boldsymbol{k} = \left\lceil \mathbf{log}_{\mathbf{2}}\, \boldsymbol{n} + \mathbf{1} \right\rceil \tag{22}$$

V předchozím vztahu představuje k použitý počet košů a n počet testovaných sekvencí. Popsaná rovnice je použita pouze v případě, že $n \geq 30$, jinak je $k = 5$.

6.1.2 SimplePresenter

Další aplikace představuje z pohledu uživatele základní nástroj pro predikování proteinových domén v předkládaných sekvencích. Na rozdíl od předchozího programu je tento založen výhradně na uživatelském rozhraní a integruje několik součástí. Mezi ně patří převod vstupní sekvence do interní reprezentace (založený na dříve zmíněném nástroji *TrainingSetConvertor*), predikce hranice s využitím neuronové sítě a samozřejmě vizuální prezentace výsledků uživateli. Výstup aplikace je doplněn o volitelnou analýzu vstupní sekvence s využitím regulárních výrazů databáze PROSITE.

Aplikace umožňuje uživateli ovlivnit výběrem vstupních dat režim své činnosti. Podporovány jsou následující možnosti.

1. Základní režim spočívá ve výběru FASTA souboru, tato varianta provádí kompletní analýzu sekvence, včetně případného použití informace z databáze PROSITE. Její nevýhodou je relativně dlouhý běh použitého nástroje PSI-BLAST (řádově několik minut v závislosti na použité databázi a délce vstupního řetězce).
2. Další varianta umožňuje vybrat pouze soubor ve vnitřní reprezentaci. Aplikace následně provádí pouze predikci hranice domény, není umožněn výpis (ani PROSITE analýza) původní FASTA sekvence. Výhodou toho režimu je vysoká rychlost jeho vykonání (řádově několik málo sekund).

3. Poslední režim předpokládá výběr souboru ve vnitřní reprezentaci a následně výběr původní FASTA sekvence. Aplikace následně provádí predikci hranice domény volitelně doplněnou o analýzu původní sekvence. Doba běhu se pohybuje kolem 30 vteřin a je stále výrazně rychlejší než u bodu 1.

Obrázek 6.5: Kompletní rozhraní nástroje pro predikci proteinových domén. Ve spodní části je zobrazen výstup neuronové sítě (původní výstup červenou křivkou a vyhlazené hodnoty modrými sloupci), který indikuje přítomnost hranice domén (označené zeleným trojúhelníkem) napříč celou analyzovanou sekvencí. Vrchní část formuláře zobrazuje především informaci o pozici predikované hranice (results) a výpis vstupní sekvence. Ve vypsané vstupní sekvenci je znázorněna predikovaná hraniční oblast červeně a příslušné hraniční residuum zeleně. Modře jsou znázorněny úseky vstupní sekvence, které odpovídají některým vzorům v databázi PROSITE. Podobného výstupu lze dosáhnou v popsaných režimech činnosti aplikace 1 a 3. V režimu 2 chybí výpis původní sekvence.

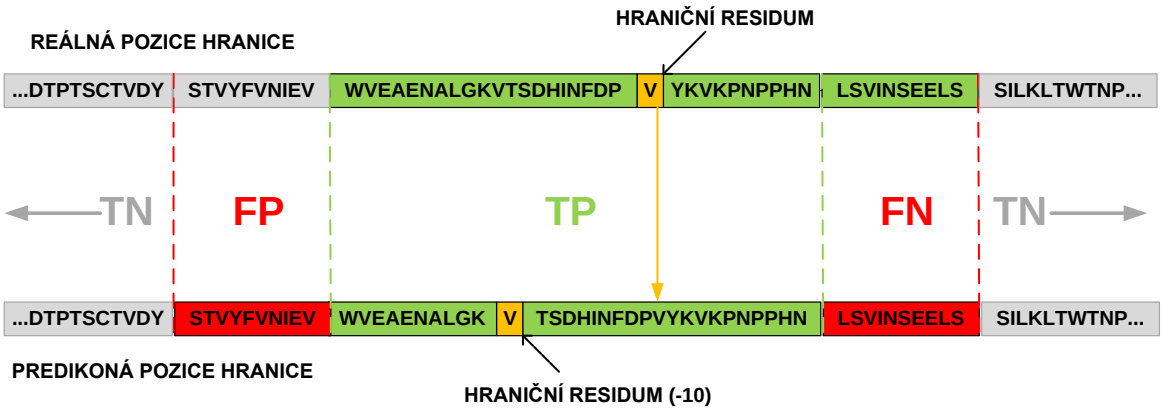
6.2 Metodika testování

V další části kapitoly se zaměřím na proces testování implementované metody, jenž jsem prováděl v návaznosti na materiál [18], ze kterého jsem vycházel při návrhu realizovaného algoritmu. Vlastnosti predikce proteinových domén jsem se rozhodl dle zmíněného materiálu ověřovat především postupem nazývaným křížová validace, vytvořeným podle následujícího schématu.

1. Ze sady 510 proteinových sekvencí, která bude následně blíže popsána, jsem náhodně vybral 50 řetězců.
2. Se zbylými 460 zástupci jsem provedl trénování neuronové sítě.
3. Po optimalizaci vah sítě jsem otestoval její úspěšnost při predikci hranice domén ve skupině sekvencí vybraných v kroku 1.
4. Postup jsem, počínaje bodem 1, opakoval celkem desetkrát. Jako výslednou úspěšnost predikce jsem následně použil vypočítanou průměrnou hodnotu ze všech běhů.

Použitá pracovní sada proteinových sekvencí byla sestavena pro trénink a testování metody PPRODO popsané v [18] a je dostupná z url: http://gene.kias.re.kr/~jlee/pprodo/set_training.tar.gz. Data jsou tvořena proteiny se dvěma souvislými doménami (tedy s jedinou hranicí) umožňující efektivní trénink a testování mnou realizované metody. Výběr použitých sekvencí vycházel ze struktur popsaných v databázi SCOP, jejichž délka musela být větší než 40 residuí (proteiny s více než jednou doménou) a současně menší než 500 residuí (byly vyloučeny vícedoménové proteiny). Tréninková sada byla navíc sestavena tak, aby byla maximální povolená sekvenční podobnost mezi libovolnou dvojicí zástupců nižší než 30 %. Celkově bylo shromážděno 522 řetězců aminokyselin, které jsem během procesu čistění dat zredukoval (kvůli neúplné hraniční oblasti) na konečných 510.

Pro ohodnocení výstupu predikce jsem použil obecně přijímané kritérium [např. 11, 15, 18], které za úspěšný považuje stav, kdy se pozice predikovaného hraničního residua nachází maximálně +/-20 aminokyselin od pozice reálné hranice. Předchozí metodiku, stejně jako interpretaci pojmů TP, FP, TN a FN v kontextu realizovaného testování shrnuje obrázek 6.6.



Obrázek 6.6: Znázornění překryvu hranice při predikci. Ve spodní části obrázku je úspěšný výstup predikce (podle dříve popsaného kritéria), jehož hraniční oblast je posunutá o -10 pozic oproti pozici reálné (zobrazené ve vrchní části obrázku). Díky tomuto posunutí lze dále ukázat správně predikovanou část hraniční oblasti označenou jako TP, nesprávně označenou hraniční oblast (FP), neoznačenou hraniční oblast (FN) a nehraniční oblast (TN). Tyto pojmy jsou použity ve výstupu dříve představeného nástroje *BPTesting*.

V návaznosti na výše popsanou metodiku jsem se rozhodl začlenit několik dodatečných testů, zaměřených na ověření schopnosti prediktoru vyrovnat se s proteiny s jedinou doménou. Zmíněné testy využívají výsledků (resp. naučených vah neuronové sítě a případně také vybrané testovací podmnožiny sekvencí) z jednotlivých běhů křížové validace k ověření schopnosti algoritmu detekovat proteiny bez hraniční oblasti (s jedinou doménou). Testování této vlastnosti jsem prováděl na základě nastavení prahové hodnoty pro výstup neuronové sítě, ve dvou krocích.

1. Pro konkrétní hodnoty prahu jsem zkoumal, v jakém počtu sekvencí z testovací sady jednodoménových proteinů (bude popsána na konci textu) je maximální výstupní hodnota prediktoru nižší než prahová (tj. kolik zástupců je označených jako 1 doménové proteiny).
2. S nastavením prahu z předchozího kroku jsem následně zkoumal schopnost prediktoru správně rozlišit počet domén v rámci proteinu. K tomuto účelu jsem použil kombinovanou testovací sadu sekvencí složenou:
    - ze sady jednodoménových proteinů využité v kroku 1,
    - ze sady dvoudoménových proteinů, použitých k testování běhu křížové validace.

Na závěr zbývá popsat původ testovací sady jednodoménových proteinů. Sada byla opět určena k testování metody PPRODO. Je volně dostupná z dříve popsané adresy a původně obsahovala 510 proteinový sekvencí, ze kterých jsem náhodně zvolil 48 zástupců (aby byl přibližně zachován poměr jedno a dvou doménových sekvencí při testu rozlišovací schopnosti).

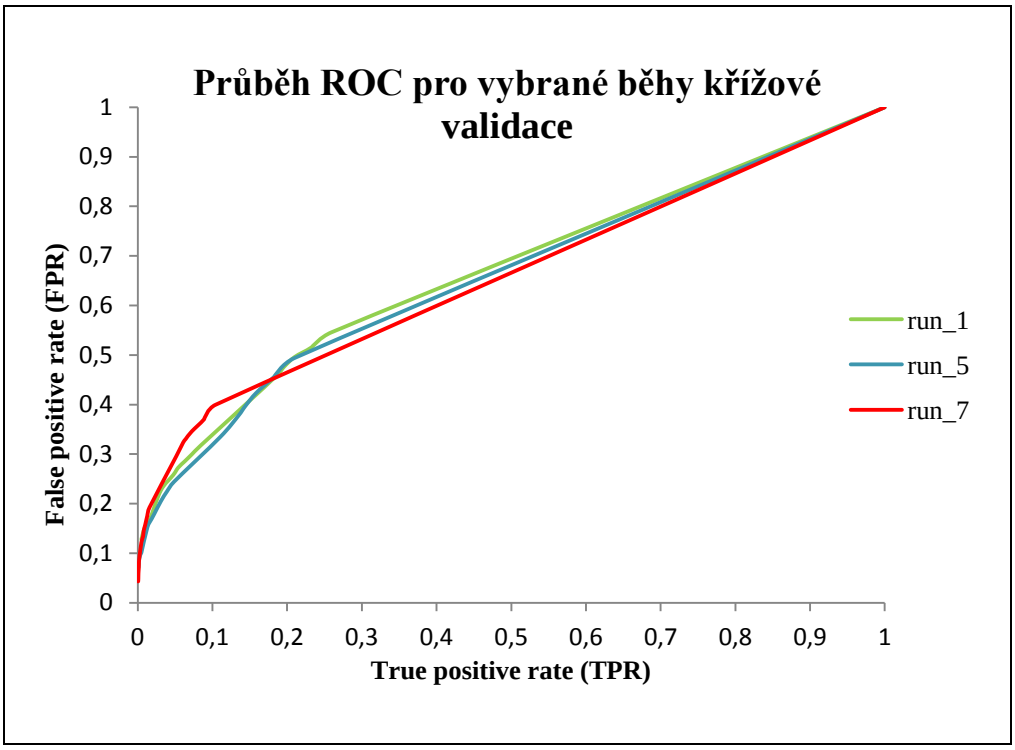
6.3 Výsledky testování

6.3.1 Výsledky křížové validace

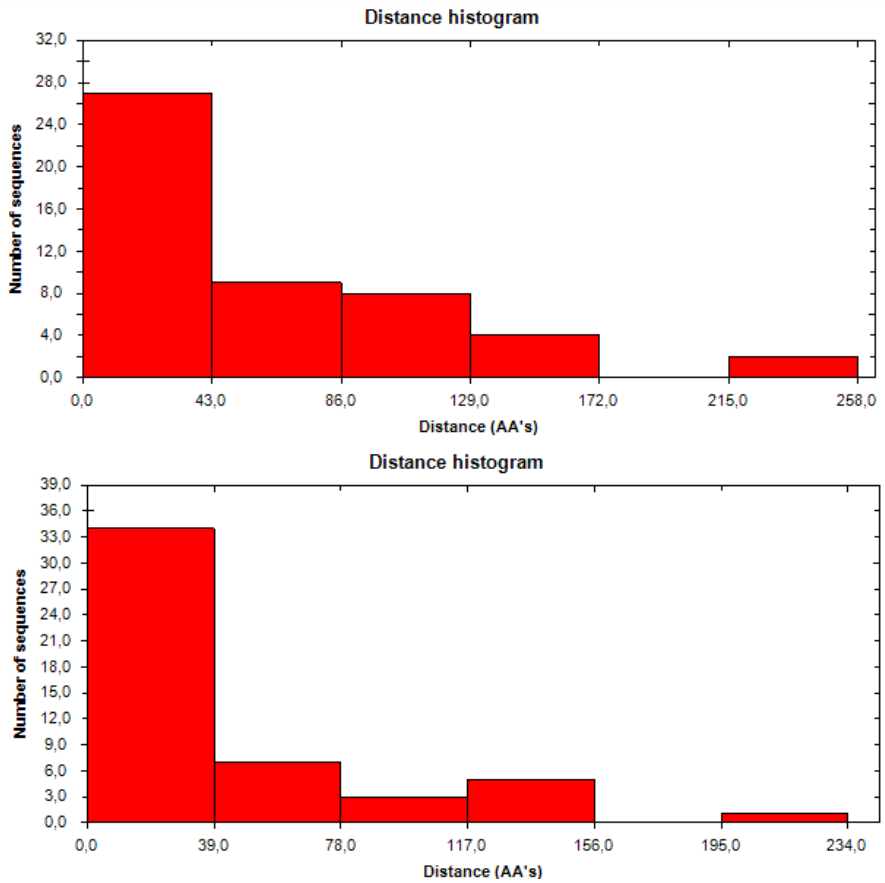
Na následujících stránkách práce jsou uvedeny výsledky běhů křížové validace, které jsem pro vyšší názornost doplnil grafem průběhů ROC křivky pro různé běhy validace, a také histogramem vzdálenosti predikovaných hranic domén od hranic reálných.

Název běhu	Iterace BP	Aplikace GA	Úspěšnost	Průměr. vzdál.
run_1	410	1	48%	58,06
run_2	365	1	50%	47,36
run_3	415	1	50%	43,88
run_4	249	2	54%	47,98
run_5	225	2	54%	50,28
run_6	205	3	56%	35,88
run_7	224	3	60%	40,98
run_8	335	4	50%	48,33
run_9	274	3	58%	55,42
run_10	205	2	48%	60,04
Průměrné hodnoty				
	290,7	**2,2**	**52,8%**	**48,86**
Minimální hodnoty				
	205	**1**	**48%**	**35,88**
Maximální hodnoty				
	415	**4**	**60%**	**60,04**

Tabulka 6.1: Výsledky deseti běhů křížové validace, včetně průměrných, maximálních a minimálních hodnot jednotlivých atributů. Sloupec *iterace BP* představuje počet iterací metody Back-Propagation nutných k nalezení nejlepšího výsledku konkrétního běhu. *Aplikace GA* představuje počet aplikací genetického algoritmu použitých k řešení lokálního extrému metody BP (1 iterace probíhá vždy k inicializaci vah neuronové sítě). Sloupec *úspěšnost* označuje procento úspěšně nalezených hranic, resp. správně predikovaných sekvencí (souborů) v daném běhu. *Průměrná vzdálenost* představuje průměrnou hodnotu vzdálenosti predikovaných hranic (resp. pozic hraničních residuí) od pozic hranic reálných.



Graf 6.1: Zobrazení průběhu ROC křivky pro nejlepší (run_7), mediánový (run_5) a nejhorší (run_5) běh křížové validace.



Obrázek 6.7: Výstup programu *BPTesting* představující histogram vzdáleností predikovaných hranic domén od hranic skutečných. Zaznamenaný pro nejlepší (horní část) a nejhorší běh křížové validace.

Název běhu	Úspěšnost	Průměr. vzdál.
run_1	96,52%	13,25
run_2	98,48%	10,95
run_3	98,91%	10,97
run_4	97,83%	11,53
run_5	96,52%	13,44
run_6	95%	13,79
run_7	96,74%	13,01
run_8	96,06%	13,05
run_9	97,17%	9,65
run_10	96,96%	11,71
Průměrné hodnoty		
	97,02%	**12,14**
Minimální hodnoty		
	95%	**9,65**
Maximální hodnoty		
	98,91%	**13,79**

Tabulka 6.2: Tabulka obsahuje výsledky testování pro datové sady, se kterými bylo *trénováno* během deseti cyklů křížové validace. Tyto informace byly začleněny, aby bylo možné porovnat vliv přeučení neuronové sítě na její schopnost zobecňovat.

V této části textu byly prezentovány výsledky testování implementovaného nástroje metodou křížové validace, tak jak byla popsána v předchozím oddílu. Průměrnou úspěšnost prediktoru jsem experimentálně stanovil na 52 %. Ze zjištěných hodnot bych chtěl objasnit poměrně vysokou hodnotu průměrné vzdálenosti mezi predikovanou hranicí a hranicí reálnou, která je způsobena vysokou odchylkou této hodnoty v rámci nesprávně predikovaných sekvencí.

6.3.2 Výsledky testování schopnosti rozlišovat jedno a dvou doménové proteiny.

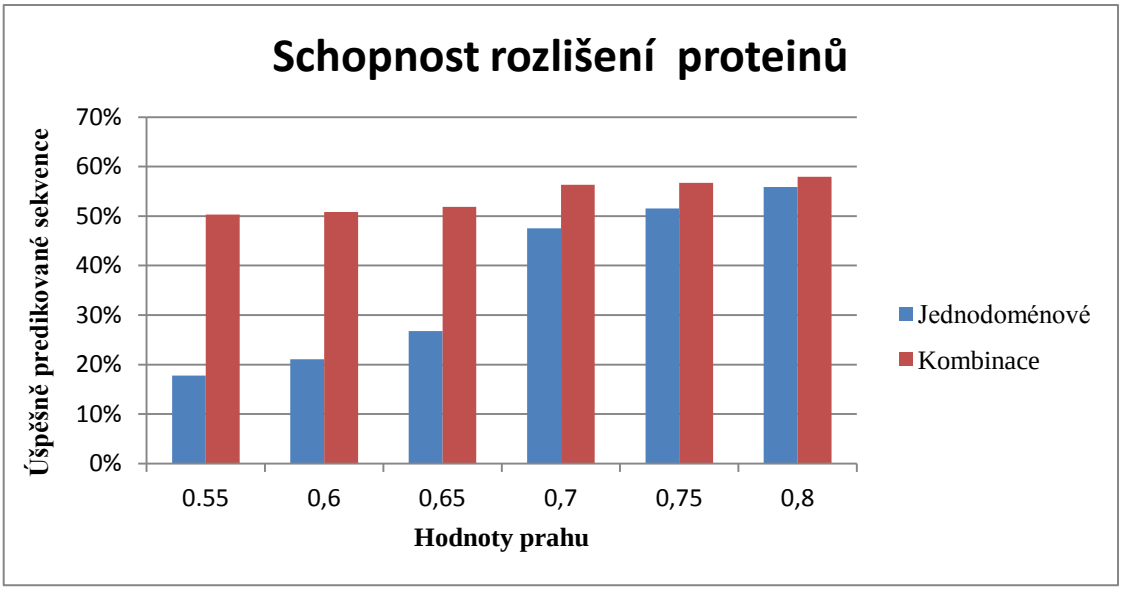
V druhé části testování jsem se zaměřil na ověření schopnosti prediktoru vyrovnat se s jednodoménovými proteiny. Při testování jsem postupoval podle metodiky popsané v předchozí kapitole. Výsledky jsou představeny ve formě dvou tabulek, které prezentují zjištěné vlastnosti realizované metody při detekci jednodoménových proteinů a schopnosti rozlišit mezi sekvencemi s jednou a dvěma doménami. Závěrečný graf pak vizualizuje zjištěné průměrné hodnoty.

Název běhu	0,55	0,6	0,65	0,7	0,75	0,8
run_1	12,5%	12,5%	14,5%	27,1%	31,3%	37,5%
run_2	18,8%	20,8%	25%	60,4%	62,5%	64,6%
run_3	10,4%	14,6%	18,75%	52%	54,2%	56,3%
run_4	10,4%	10,4%	16,7%	39,6%	41,7%	47,9%
run_5	29,2%	37,5%	37,5%	52,1%	56,3%	62,5%
run_6	27,1%	27,1%	31,3%	47,9%	47,9%	54,17%
run_7	33,3%	41,7%	43,8%	62,5%	77,1%	77,1%
run_8	14,6%	14,6%	16,7%	37,5%	37,5%	54,2%
run_9	13,4%	19,4%	34%	50,2%	63,1%	64,7%
run_10	20,8%	25%	43,8%	72,9%	75%	77,1%
Průměrné hodnoty						
	17,8%	**21,1%**	**26,8%**	**47,5%**	**51,5%**	**55,9%**
Minimální hodnoty						
	10,4%	**10,4%**	**14,5%**	**27,1%**	**31,3%**	**37,5%**
Maximální hodnoty						
	33,3%	**41,7%**	**43,8%**	**72%**	**77,1%**	**77,1%**

Tabulka 6.3: Tabulka obsahuje výsledky testování schopnosti algoritmu *rozeznat jednodoménové proteiny*. K testování je využita dříve popsaná sada jednodoménových proteinů a váhy neuronové sítě vypočítané během křížové validace. Výstup prediktoru je testován pro různé hodnoty prahu, které indikuje vrchní řádek tabulky. Hodnoty uvnitř tabulky pak označují procento správně zpracovaných souborů (pro které je maximální výstup predikce nižší než konkrétní prahová hodnota) v jednodoménové sadě dat.

Název běhu	0,55	0,6	0,65	0,7	0,75	0,8
run_1	53,1%	53,1%	54,1%	54,1%	56,1%	59,2%
run_2	51%	52%	53,1%	60,2%	59,2%	60,2%
run_3	45,9%	48%	50%	55,1%	53,1%	53,1%
run_4	51%	51%	53,1%	58,2%	58,2%	60,2%
run_5	57,1%	57,1%	56,1%	59,2%	60,2%	57,1%
run_6	49%	48%	51%	52%	49%	51%
run_7	50%	53,1%	52%	57,1%	62,2%	59,2%
run_8	52%	51%	50%	55,1%	54,1%	60,2%
run_9	50%	50%	49%	55,1%	57,1%	61,2%
run_10	43,9%	44,9%	50%	57,1%	58,2%	58,2%
Průměrné hodnoty						
	50,3%	**50,8%**	**51,8%**	**56,3%**	**56,4%**	**57,6%**
Minimální hodnoty						
	43,9%	**44,9%**	**44,9%**	**52%**	**49%**	**51%**
Maximální hodnoty						
	57,1%	**57,1%**	**56,1%**	**60,2%**	**62,2%**	**61,2%**

Tabulka 6.3: Tabulka obsahuje výsledky testování schopnosti algoritmu *rozlišit jedno a dvou doménové proteiny*. K testování je využita dříve popsaná kombinovaná sada proteinů a váhy neuronové sítě vypočítané během křížové validace. Výstup prediktoru je testován pro různé hodnoty prahu, které indikuje vrchní řádek tabulky. Hodnoty uvnitř tabulky pak označují procento správně přiřazených typů proteinů v kombinované sadě dat. Při tom se kontroluje maximální výstup prediktoru, který určí, jestli je analyzovaný protein jedno doménový (hodnota je nižší než konkrétní práh), nebo dvojdoménový (hodnota je vyšší nebo rovna prahu).



Graf 6.2: Graf vizualizuje výsledky popsané předchozími tabulkami. Konkrétně úspěšnost, s jakou pro jednotlivé prahové hodnoty prediktor správně určí jednorozměrné proteiny v testovací sadě jednorozměrných proteinů (modré sloupce), a také úspěšnost, s jakou je prediktor schopen rozlišovat (pro zmíněné prahové hodnoty) mezi jedno a dvou doménovými proteiny v kombinované datové sadě (červené sloupce).

Během testování schopnosti algoritmu rozlišovat jedno a dvou doménové proteiny se mi podařilo přibližně určit, které hodnoty prahu tuto vlastnost nejlépe podporují. Podle zjištěných výsledků bych k vyloučení jednodoménových proteinů doporučil výstup prediktoru prahovat hodnotou 0,7. Takové nastavení představuje nejlepší kompromis mezi dosaženými minimálními, maximálními a průměrnými hodnotami úspěšného rozlišení jedno a dvou doménových proteinů.

6.4 Náměty pro další rozvoj

Poslední část kapitoly bych chtěl věnovat možným budoucím cestám rozvoje této aplikace. Prvním bezesporu účelným vylepšení by mohlo být začlenění predikovaných informací o sekundární struktuře proteinu a přístupnosti rozpouštědlu. Obě zmíněné složky by vhodně doplňovali použitou PSSM. Podle mého názoru by ideální formou jejich integrace mohla být metoda popsaná v [18]. Kde je použita dvojice neuronových sítí. První zpracovává evoluční informaci obsaženou ve vstupním profilu a výstup její predikce slouží v kombinaci s výše zmíněnými informacemi jako vstup druhé sítě, která provádí konečnou predikci.

V návaznosti na předchozí krok by mohla být rozumným rozšířením úprava finální fáze predikce. Kde by ideálně zůstalo zachováno prvotní vyhlazení výstupu neuronové sítě, ale namísto původního přiřazení hranice domény na základě vyhledání maxima ve vyhlazeném výstupu, by se dal využit přístup popsaný v kapitole o algoritmu DOMpro. Tento vyhledává hraniční oblast ve výstupu

predikce pomocí shody s definovaným vzorem podobným regulárnímu výrazu. Právě popsané rozšíření by umožňovalo algoritmus použít také k predikci hranic domén v rámci proteinů s libovolným množstvím souvislých struktur.

Velmi zajímavým prvkem, který by určitě výrazně vylepšil kvalitu výsledné predikce, by mohla být specificky sestavená tréninková sada, která by v ideálním případě měla obsahovat příklady hranic, pro veškeré kombinace dvojic známých domén (zastoupených např. v databázi CATH). Sestavení podobné sady sekvencí by ale bylo samo o sobě velmi náročným úkolem. Navíc by mohl charakter vzniklých dat vést k nutnosti dalších úprav realizované metody, například k hierarchické aplikaci několika klasifikátorů zaměřených na různé podmnožiny řetězců (vlivem protichůdných vzorů v rámci datové sady apod.) a jejího vyhodnocení (příkladem oprávněnosti této úvahy může být metoda DOBO, popsaná v předchozím textu).

Posledním místem, kde by se dala práce ještě vylepšovat, by mohla být popsaná aplikace genetického algoritmu pro korekci běhu metody Back-Propagation. Kde v současné podobě chybí objektivnější systematické hodnocení jejího přínosu.

7 Závěr

V práci jsem nastínil oblast proteinů a jejich struktury. V rámci krátké pasáže jsem představil základní stavební jednotky těchto molekul – aminokyseliny. Dále jsem zmínil vybrané laboratorní postupy vedoucí k získání informací o struktuře proteinů na různých úrovních hierarchie. Následně jsem se pokusil definovat pojem proteinové domény včetně její strukturní, fyzické a evoluční funkce, a také některé způsoby klasifikace těchto útvarů.

Poměrně významný prostor jsem věnoval existujícím nástrojům pracujícím na principu predikce proteinových domén nebo využívajícím její výsledky. Mezi tyto nástroje jsem zařadil také několik významných zástupců veřejně dostupných databází proteinových domén a především webové servery umožňující anotovat předkládanou sekvenci aminokyselin. V rámci jednotlivých nástrojů jsem se zaměřoval na charakter metod jimi využitých, na funkce, které nabízejí uživateli, a pro ilustraci jsem navíc připojil náhled jejich uživatelského rozhraní. Abych ověřil funkci zde popsaných metod, prováděl jsem s nimi jednoduchý experiment, během něhož jsem je nechal zpracovat konkrétní sekvenci aminokyselin.

Na výše zmíněnou část jsem navázal studiem vybraných postupů predikce. Při volbě konkrétních zástupců jsem se snažil zachovat historický vývoj tohoto odvětví výzkumu, proto jsem nejprve analyzoval několik, podle mého názoru, významných metod pracujících s vnitřní strukturou molekul a popsal univerzální princip, který při své činnosti obvykle užívají. Následně jsem se zaměřil na mladší přístupy využívající ke své činnosti primární strukturu proteinu v kombinaci s metodami strojového učení a opět jsem se pokusil identifikovat centrální úvahu, ze které vychází.

Poslední část práce jsem věnoval rozboru realizované aplikace predikující domény z primární struktury proteinu. Implementace vychází z obecných principů metod stejného typu popsaných v této práci a především z materiálu [18]. Vytvořený algoritmus se snaží vyhledat vzory specifických útvarů známých jako hranice proteinových domén v profilu multisekvenčního zarovnání metody PSI-BLAST s využitím dopředné neuronové sítě. Mezi potenciální přínosy rozšiřující výchozí materiál [18] bych zařadil především dříve popsané úpravy korigující běh algoritmu Back-Propagation.

V závěru textu jsem se zabýval testováním vytvořeného nástroje metodou křížové validace. K určení úspěšnosti predikce jsem použil obecně přijímané kritérium, definující predikci jako úspěšnou, pokud leží přepokládané hraniční residuum ve vzdálenosti +/- 20 pozic od reálného umístění hranice. Pro rozšíření výchozího testování jsem dále zjišťoval, jak je metoda schopná rozlišit proteiny s jednou a dvěma doménami. Výsledky testování a náměty pro případný další rozvoj práce jsou uvedeny v příslušných kapitolách.

Literatura

[1] ALBERTS, Bruce. *Základy buněčné biologie: Úvod do molekulární biologie buňky*. 2. vyd. Překlad Arnošt Kotyk, Bohumil Bouzek, Pavel Hozák. Ústí nad Labem: Espero, 2005, xxvi, 630, D-18, A-62, I-30 s. ISBN 80-902-9062-0.

[2] ŘEHULKA, Pavel. *Základy bioinformatického zpracování dat v proteomice*. Fakulta vojenského zdravotnictví UO, 2010. Dostupné z: http://www.pmfhk.cz/WWW/UMP/aplikovana_proteomika/bioinformatika_pr_cv.pdf

[3] *Protein structure* [online], poslední aktualizace 8. ledna 2013 16:54 [cit. 8. 1. 2013], Wikipedie. Dostupné z WWW: http://en.wikipedia.org/w/index.php?title=Protein_structure&action=history

[4] REYNOLDS, Christopher. SCHOOL OF BIOLOGICAL SCIENCES. *Domain swapping in G-protein coupled receptor dimers* [online]. 2002 [cit. 2013-01-09]. Dostupné z: http://www.essex.ac.uk/bs/bs_staff/reynolds/dimer.htm

[5] *Protein domain* [online], poslední aktualizace 6. ledna 2013 00:50 [cit. 8. 1. 2013], Wikipedie. Dostupné z WWW: http://en.wikipedia.org/wiki/Protein_domain

[6] MURZIN, Alex, Steven BRENNER a Tim HUBBARD. SCOP: A Structural Classification of Proteins Database. *JMB* [online]. 1994, č. 247, s. 5 [cit. 2013-01-09]. ISSN CAM 502/94. Dostupné z: http://www.osti.gov/eprints/topicpages/documents/record/041/1226608.html

[7] CONTE, Loredana, Bart AILEY a Tim HUBBARD. SCOP: a Structural Classification of Proteins database. *Nucleic Acids Research* [online]. 2000, č. 28 [cit. 2013-01-09]. Dostupné z: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC102479/

[8] The Pfam Protein Families Database. *Nucleic Acids Research* [online]. 2002, č. 30 [cit. 2013-01-09]. Dostupné z: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC99071/

[9] FINN, Robert, Jaina MISTRY a John TATE. The Pfam protein families database. *Nucleic Acids Research* [online]. 2010, č. 38, s. 12 [cit. 2013-01-09]. Dostupné z: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2808889/

[10] *Current Protein and Peptide Science* [online]. 2007-04-01, roč. 8, č. 2 [cit. 2013-01-09]. ISSN 13892037. Dostupné z: http://openurl.ingenta.com/content/xref?genre=article

[11] VERETNIK, Stella, Jenny GU a Soshana WODAK. *Structural bioinformatics: IDENTIFYING STRUCTURAL DOMAINS*. 2nd ed. Hoboken, N.J.: Wiley-Blackwell, c2009. ISBN 978-0-470-18105-8.

[12] 2Can Support Portal - Protein Function: InterProScan - Introduction. EMBL-EBI. *EMBL-EBI* [online]. 2010 [cit. 2013-01-09]. Dostupné z: http://www.ebi.ac.uk/2can/tutorials/function/InterProScan.html

[13] 2Can Support Portal - Protein Function: InterProScan - Introduction. EMBL-EBI. *EMBL-EBI* [online]. 2010 [cit. 2013-01-09]. Dostupné z: http://www.ebi.ac.uk/2can/tutorials/function/InterProScan2.html

[14] EICKHOLT, Jesse, Xin DENG a Jianlin CHENG. DoBo: Protein domain boundary prediction by integrating evolutionary signals and machine learning. *BMC Bioinformatics* [online]. 2011, roč. 12, č. 1, s. 43- [cit. 2013-01-09]. ISSN 1471-2105. DOI: 10.1186/1471-2105-12-43. Dostupné z: http://www.biomedcentral.com/1471-2105/12/43

[15] HOLM, Liisa a Chris SANDER. Parser for Protein Folding Units. *PROTEINS: Structure, Function, and Genetics*. 1994, č. 19. Dostupné z: http://pdomains.sdsc.edu/v2/puu.php

[16] Continuous and discontinuous domain: An algorithm for the automatic generation. *Protein Science* [online]. 1995, č. 4 [cit. 2013-01-09]. Dostupné z: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2143117/

[17] CRIPPEN, Gordon M. Recognizing protein folds by cluster distance geometry. *Proteins: Structure, Function, and Bioinformatics* [online]. 2005-07-01, roč. 60, č. 1, s. 82-89 [cit. 2013-01-09]. ISSN 08873585. DOI: 10.1002/prot.20488. Dostupné z: http://doi.wiley.com/10.1002/prot.20488

[18] SIM, Jaehyun, Seung-Yeon KIM a Jooyoung LEE. PPRODO: Prediction of protein domain boundaries using neural networks. *Proteins: Structure, Function, and Bioinformatics* [online]. 2005-05-15, roč. 59, č. 3, s. 627-632 [cit. 2013-01-09]. ISSN 08873585. DOI: 10.1002/prot.20442. Dostupné z: http://doi.wiley.com/10.1002/prot.20442

[19] CHENG, Jianlin, Michael J. SWEREDOSKI a Pierre BALDI. DOMpro: Protein Domain Prediction Using Profiles, Secondary Structure, Relative Solvent Accessibility, and Recursive Neural Networks. *Data Mining and Knowledge Discovery* [online]. 2006-6-12, roč. 13, č. 1, s. 1-10 [cit. 2013-01-09]. ISSN 1384-5810. DOI: 10.1007/s10618-005-0023-5. Dostupné z: http://www.springerlink.com/index/10.1007/s10618-005-0023-5

[20] CHAN, L.-W. a F. FALLSIDE. An adaptive training algorithm for back propagation networks. *Computer Speech* [online]. 1987, roč. 2, 3-4, s. 205-218 [cit. 2013-01-09]. ISSN 08852308. DOI: 10.1016/0885-2308(87)90009-X. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/088523088790009X

[21] HILL, M. O. Reciprocal Averaging: An Eigenvector Method of Ordination. *The Journal of Ecology*. 1973, roč. 61, č. 1, s. 12. Dostupné z: http://www.jstor.org/stable/2258931

[22] XU, Y., D. XU a H. N. GABOW. Protein domain decomposition using a graph-theoretic approach. *Bioinformatics*. 2000, roč. 16, č. 12, s. 1091-1104. ISSN 1367-4803. DOI: 10.1093/bioinformatics/16.12.1091. Dostupné z: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/16.12.1091

[23] MADEJ, Thomas, Jean-Francois GIBRAT a Stephen H. BRYANT. Threading a database of protein cores. *Proteins: Structure, Function, and Genetics*[online]. 1995, roč. 23, č. 3, s. 356-369 [cit. 2013-04-30]. ISSN 0887-3585. DOI: 10.1002/prot.340230309. Dostupné z: http://doi.wiley.com/10.1002/prot.340230309

[24] BALDI, Pierre a Gianluca POLLASTRI. The Principled Design of Large-Scale Recursive Neural Network Architectures–DAG-RNNs and the Protein Structure Prediction Problem. *Journal of Machine Learning Research* [online]. 2003, č. 4, s. 575-602 [cit. 2013-05-02]. Dostupné z: http://www.ai.mit.edu/projects/jmlr/papers/volume4/baldi03a/source/baldi03a.pdf

[25] Exon shuffling. CONTRIBUTORS, Wikipedia. WIKIPEDIA, The Free Encyclopedia. Wikipedia: The Free Encyklopedia [online]. 2013, 28.5.2013 [cit. 2013-05-07]. Dostupné z: http://en.wikipedia.org/w/index.php?title=Exon_shuffling&oldid=552512659

[26] ALTSCHUL, S. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research* [online]. 1997, vol. 25, issue 17, s. 3389-3402 [cit. 2013-05-09]. DOI: 10.1093/nar/25.17.3389. Dostupné z: http://www.nar.oupjournals.org/cgi/doi/10.1093/nar/25.17.3389

[27] ZBOŘIL, František. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ, Fakulta informačních technologií. *Biologický a umělý neuron, neuronové sítě. Perceptron, Adaline, Madaline a BP (Back Propagation)*. Brno, 2013. Dostupné z: https://wis.fit.vutbr.cz/FIT/st/course-files-st.php/course/SFC-IT/lectures/10sfc_2.pdf?cid=8854

[28] ZBOŘIL, František. VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ, Fakulta informačních technologií. *Genetické algoritmy*. Brno, 2013. Dostupné z: https://wis.fit.vutbr.cz/FIT/st/course-files-st.php/course/SFC-IT/lectures/06sfc_9.pdf?cid=8854

[29] WikiSkripta. *ROC křivka* [online]. 2013 [cit. 2013-05-14]. Dostupné z: http://www.wikiskripta.eu/index.php/ROC_k%C5%99ivka

[30] *Histogram* [online], poslední aktualizace 13. duben 2013 12:33 [cit. 14. 5. 2013], Wikipedie. Dostupné z WWW: http://en.wikipedia.org/w/index.php?title=Histogram&oldid=554877977

Seznam příloh

Příloha1	CD s elektronickou verzí technické zprávy
Příloha 2	CD se zdrojovými texty implementované metody
Příloha 3	Manuál pro instalaci a překlad aplikace
Příloha 4	Manuál k vytvořeným nástrojům